

Complexity

Neural Network for Complex Systems: Theory and Applications

Lead Guest Editor: Chenguang Yang

Guest Editors: Jing Na, Guang Li, Yanan Li, and Junpei Zhong





Neural Network for Complex Systems: Theory and Applications



Neural Network for Complex Systems: Theory and Applications

Lead Guest Editor: Chenguang Yang

Guest Editors: Jing Na, Guang Li, Yanan Li, and Junpei Zhong



Copyright © 2018 Hindawi. All rights reserved.

This is a special issue published in “Complexity.” All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Editorial Board

José Ángel Acosta, Spain
Rodrigo Aldecoa, USA
Juan A. Almendral, Spain
David Arroyo, Spain
Mohamed Boutayeb, France
Arturo Buscarino, Italy
Guido Caldarelli, Italy
Giulio Cimini, Italy
Danilo Comminiello, Italy
Manlio De Domenico, Italy
Pietro De Lellis, Italy
Albert Diaz-Guilera, Spain
Thach Ngoc Dinh, France
Jordi Duch, Spain
Marcio Eisencraft, Brazil
Joshua Epstein, USA
Thierry Floquet, France

Mattia Frasca, Italy
Lucia Valentina Gambuzza, Italy
Bernhard C. Geiger, Austria
Carlos Gershenson, Mexico
Peter Giesl, UK
Sergio Gómez, Spain
Sigurdur F. Hafstein, Iceland
Giacomo Innocenti, Italy
Mahdi Jalili, Australia
Jeffrey H. Johnson, UK
Vincent Labatut, France
Xiaoping Liu, Canada
Vittorio Loreto, Italy
Didier Maquin, France
Eulalia Martínez, Spain
Ch. P. Monterola, Philippines
Roberto Natella, Italy

Irene Otero-Muras, Spain
Daniela Paolotti, Italy
Luis M. Rocha, USA
Miguel Romance, Spain
Matilde Santos, Spain
Hiroki Sayama, USA
Michele Scarpiniti, Italy
Enzo Pasquale Scilingo, Italy
Dan Selișteanu, Romania
Samuel Stanton, USA
Roberto Tonelli, Italy
Shahadat Uddin, Australia
Gaetano Valenza, Italy
Dimitri Volchenkov, USA
Christos Volos, Greece

Contents

Neural Network for Complex Systems: Theory and Applications

Chenguang Yang , Jing Na, Guang Li, Yanan Li, and Junpei Zhong
Editorial (2 pages), Article ID 3141805, Volume 2018 (2018)

Visual Semantic Navigation Based on Deep Learning for Indoor Mobile Robots

Li Wang, Lijun Zhao , Guanglei Huo, Ruifeng Li, Zhenghua Hou, Pan Luo, Zhenye Sun, Ke Wang,
and Chenguang Yang 
Research Article (12 pages), Article ID 1627185, Volume 2018 (2018)

Parallel Excitatory and Inhibitory Neural Circuit Pathways Underlie Reward-Based Phasic Neural Responses

Huanyuan Zhou, KongFatt Wong-Lin, and Da-Hui Wang 
Research Article (20 pages), Article ID 4356767, Volume 2018 (2018)

Adaptive Neural Networks Control Using Barrier Lyapunov Functions for DC Motor System with Time-Varying State Constraints

Lei Ma and Dapeng Li 
Research Article (9 pages), Article ID 5082401, Volume 2018 (2018)

Application of Federal Kalman Filter with Neural Networks in the Velocity and Attitude Matching of Transfer Alignment

Lijun Song , Zhongxing Duan , Bo He, and Zhe Li
Research Article (7 pages), Article ID 3039061, Volume 2018 (2018)

Gender and Handedness Prediction from Offline Handwriting Using Convolutional Neural Networks

Ángel Morera, Ángel Sánchez , José Francisco Vélez, and Ana Belén Moreno
Research Article (14 pages), Article ID 3891624, Volume 2018 (2018)

Dynamic Traffic Congestion Simulation and Dissipation Control Based on Traffic Flow Theory Model and Neural Network Data Calibration Algorithm

Li Wang, Shimin Lin, Jingfeng Yang, Nanfeng Zhang, Ji Yang, Yong Li, Handong Zhou, Feng Yang,
and Zhifu Li
Research Article (11 pages), Article ID 5067145, Volume 2017 (2018)

Fault Diagnosis Method of Check Valve Based on Multikernel Cost-Sensitive Extreme Learning Machine

Jun Ma, Jiande Wu, and Xiaodong Wang
Research Article (19 pages), Article ID 8395252, Volume 2017 (2018)

RBF Nonsmooth Control Method for Vibration of Building Structure with Actuator Failure

Jianhui Wang, Chunliang Zhang, Houyao Zhu, Xiaofang Huang, and Li Zhang
Research Article (7 pages), Article ID 2513815, Volume 2017 (2018)

Structure Optimization of a Vibration Suppression Device for Underwater Moored Platforms Using CFD and Neural Network

Zhaoyong Mao and Fuliang Zhao
Research Article (21 pages), Article ID 5392539, Volume 2017 (2018)

Layout Optimization of Two Autonomous Underwater Vehicles for Drag Reduction with a Combined CFD and Neural Network Method

Wenlong Tian, Zhaoyong Mao, Fuliang Zhao, and Zhicao Zhao
Research Article (15 pages), Article ID 5769794, Volume 2017 (2018)

Skill Learning for Intelligent Robot by Perception-Action Integration: A View from Hierarchical Temporal Memory

Xinzheng Zhang, Jianfen Zhang, and Junpei Zhong
Research Article (16 pages), Article ID 7948684, Volume 2017 (2018)

Adaptive Neural Network Control of Serial Variable Stiffness Actuators

Zhao Guo, Yongping Pan, Tairen Sun, Yubing Zhang, and Xiaohui Xiao
Research Article (9 pages), Article ID 5361246, Volume 2017 (2018)

Semiactive Nonsmooth Control for Building Structure with Deep Learning

Qing Wang, Jianhui Wang, Xiaofang Huang, and Li Zhang
Research Article (8 pages), Article ID 6406179, Volume 2017 (2018)

Neural Learning Control of Flexible Joint Manipulator with Predefined Tracking Performance and Application to Baxter Robot

Min Wang, Huiping Ye, and Zhiguang Chen
Research Article (14 pages), Article ID 7683785, Volume 2017 (2018)

A Brief Review of Neural Networks Based Learning and Control and Their Applications for Robots

Yiming Jiang, Chenguang Yang, Jing Na, Guang Li, Yanan Li, and Junpei Zhong
Review Article (14 pages), Article ID 1895897, Volume 2017 (2018)

The Hierarchical Iterative Identification Algorithm for Multi-Input-Output-Error Systems with Autoregressive Noise

Jiling Ding
Research Article (11 pages), Article ID 5292894, Volume 2017 (2018)

Applying Two-Stage Neural Network Based Classifiers to the Identification of Mixture Control Chart Patterns for an SPC-EPC Process

Yuehjen E. Shao, Po-Yu Chang, and Chi-Jie Lu
Research Article (10 pages), Article ID 2323082, Volume 2017 (2018)

A Gain-Scheduling PI Control Based on Neural Networks

Stefania Tronci and Roberto Baratti
Research Article (8 pages), Article ID 9241254, Volume 2017 (2018)

Modeling and Error Compensation of Robotic Articulated Arm Coordinate Measuring Machines Using BP Neural Network

Guanbin Gao, Hongwei Zhang, Hongjun San, Xing Wu, and Wen Wang
Research Article (8 pages), Article ID 5156264, Volume 2017 (2018)

Adaptive Neural Network Control for Nonlinear Hydraulic Servo-System with Time-Varying State Constraints

Shu-Min Lu and Dong-Juan Li

Research Article (11 pages), Article ID 6893521, Volume 2017 (2018)

Forecasting the Acquisition of University Spin-Outs: An RBF Neural Network Approach

Weiwei Liu, Zhile Yang, and Kexin Bi

Research Article (8 pages), Article ID 6920904, Volume 2017 (2018)

A Novel SHLNN Based Robust Control and Tracking Method for Hypersonic Vehicle under Parameter Uncertainty

Chuanfeng Li, Hao Wu, Zhile Yang, Yongji Wang, and Zeyu Sun

Research Article (14 pages), Article ID 6034786, Volume 2017 (2018)

Intelligent Image Recognition System for Marine Fouling Using Softmax Transfer Learning and Deep Convolutional Neural Networks

C. S. Chin, JianTing Si, A. S. Clare, and Maode Ma

Research Article (9 pages), Article ID 5730419, Volume 2017 (2018)

Adaptive Neural Network Sliding Mode Control for Quad Tilt Rotor Aircraft

Yanchao Yin, Hongwei Niu, and Xiaobao Liu

Research Article (13 pages), Article ID 7104708, Volume 2017 (2018)

The Dissolved Oxygen Prediction Method Based on Neural Network

Zhong Xiao, Lingxi Peng, Yi Chen, Haohuai Liu, Jiaqing Wang, and Yangang Nie

Research Article (6 pages), Article ID 4967870, Volume 2017 (2018)

Stability Analysis of Impulsive Stochastic Reaction-Diffusion Cellular Neural Network with Distributed Delay via Fixed Point Theory

Ruofeng Rao and Shouming Zhong

Research Article (9 pages), Article ID 6292597, Volume 2017 (2018)

Multiconstrained Network Intensive Vehicle Routing Adaptive Ant Colony Algorithm in the Context of Neural Network Analysis

Shaopei Chen, Ji Yang, Yong Li, and Jingfeng Yang

Research Article (9 pages), Article ID 8594792, Volume 2017 (2018)

Neural Network-Based State Estimation for a Closed-Loop Control Strategy Applied to a Fed-Batch Bioreactor

Santiago Rómoli, Mario Serrano, Francisco Rossomando, Jorge Vega, Oscar Ortiz, and Gustavo Scaglia

Research Article (16 pages), Article ID 9391879, Volume 2017 (2018)

Dynamic Learning from Adaptive Neural Control of Uncertain Robots with Guaranteed Full-State Tracking Precision

Min Wang, Yanwen Zhang, and Huiping Ye

Research Article (14 pages), Article ID 5860649, Volume 2017 (2018)

Composite Learning Sliding Mode Control of Flexible-Link Manipulator

Bin Xu and Pengchao Zhang

Research Article (6 pages), Article ID 9430259, Volume 2017 (2018)

Multisynchronization for Coupled Multistable Fractional-Order Neural Networks via Impulsive Control

Jin-E Zhang

Research Article (10 pages), Article ID 9323172, Volume 2017 (2018)

Adaptive Neural Output Feedback Control for Uncertain Robot Manipulators with Input Saturation

Rong Mei and Chengjiang Yu

Research Article (12 pages), Article ID 7413642, Volume 2017 (2018)

General Recurrent Neural Network for Solving Generalized Linear Matrix Equation

Zhan Li, Hong Cheng, and Hongliang Guo

Research Article (7 pages), Article ID 9063762, Volume 2017 (2018)

Minimal-Learning-Parameter Technique Based Adaptive Neural Sliding Mode Control of MEMS Gyroscope

Bin Xu and Pengchao Zhang

Research Article (8 pages), Article ID 6019175, Volume 2017 (2018)

Pinning Synchronization for Complex Networks with Interval Coupling Delay by Variable Subintervals Method and Finsler's Lemma

Dawei Gong, Frank L. Lewis, Liping Wang, Dong Dai, and Shuang Zhang

Research Article (9 pages), Article ID 2137103, Volume 2017 (2018)

Editorial

Neural Network for Complex Systems: Theory and Applications

Chenguang Yang ¹, Jing Na,² Guang Li,³ Yanan Li,⁴ and Junpei Zhong⁵

¹College of Automation Science and Engineering, South China University of Technology, Guangzhou, China

²Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming, China

³School of Engineering and Materials Science, Queen Mary University of London, London, UK

⁴Department of Engineering and Design, University of Sussex, Brighton, UK

⁵National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Correspondence should be addressed to Chenguang Yang; cyang@ieee.org

Received 7 March 2018; Accepted 13 March 2018; Published 8 May 2018

Copyright © 2018 Chenguang Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Over the last few decades, neural network (NN) has been successfully employed in a wide range of applications as the joint effort from both industrial and academic communities. With its powerful approximation ability, NN has been utilized into many promising fields, such as modelling and identification of complex and nonlinear systems and optimization and automatic control. The components integrated with a complex system may interact with each other and bring difficulties to the control. Hence, research of the complex system is becoming increasingly important in both the natural and social sciences. In this special issue, we bring together a number of important works. The contributors have investigated various applications of a wide range of complex systems such as robots and transportation systems and have also achieved a number of advances in fundamental theoretical studies.

In the field of robotics, Y. Jiang et al. investigated and surveyed the state of the art on NN based robot learning and control applications. In this work, the authors summarized recent progresses in neural networks and their applications in robots, for example, NN based robot manipulator control, NN based human robot interaction, and NN based behaviour recognition and generation. G. Gao et al. proposed a modelling and error compensation method for articulated arm coordinate measuring machine (AACMM) based on BP neural network, which shows that 97% error of the AACMM can be removed after compensation. Z. Guo et al. derived a multi-input multi-output complex nonlinear dynamic model to fully describe serial variable stiffness actuators (SVSAs)

and proposed an NN-based adaptive control strategy based on feedback linearization to handle system uncertainties. R. Mei and C. Yu presented an adaptive neural output feedback control scheme for uncertain robot manipulators with input saturation using the radial basis function neural network (RBFNN) and disturbance observer. M. Wang et al. developed a dynamic learning method for an uncertain n -link robot with unknown system dynamics and achieved predefined performance attributes on the link angular position and velocity tracking errors. In another work, M. Wang et al. focused on neural learning from the adaptive neural control (ANC) for a class of flexible joint manipulator with unknown dynamics under the output tracking error constraint. X. Zhang et al. presented a skill learning method by perception-action integration strategy from the perspective of hierarchical temporal memory (HTM) theory. B. Xu and P. Zhang proposed the sliding mode control while the adaptive design was developed using neural networks (NNs) and disturbance observer (DOB) with novel update laws for NN and DOB. L. Wang et al. proposed a three-layer perception framework based on transfer learning which improved the environmental perception ability of mobile robots during semantic navigation.

On complex system control issues, L. Song et al. used the federal Kalman Filter (FKF) based on NN in the velocity and attitude matching of transfer alignment. L. Ma and D. Li proposed an adaptive NN control approach for a direct-current (DC) system with full state constraints. J. Ding used the hierarchical identification principle to present a

hierarchical gradient based iterative (H-GI) algorithm and a hierarchical least square based iterative (H-LSI) algorithm. B. Xu and P. Zhang investigated an adaptive neural sliding mode controller for MEMS gyroscopes with minimal-learning-parameter technique. C. Li et al. proposed a combined system modelling approach to approximate the actual vehicle system. S.-M. Lu and D.-J. Li addressed an adaptive NN control problem for a class of nonlinear hydraulic servosystem with time-varying state constraints. Y. Yin et al. proposed a novel NN sliding mode control based on a multicommunity bidirectional drive collaborative search algorithm (M-CBDCS) to design a flight controller for performing the attitude tracking control of a quad tilt rotors aircraft (QTRA). Z. Li et al. proposed a general framework of a nonlinear recurrent NN for online solving generalized linear matrix equation (GLME) with global convergence property. S. Tronci and R. Baratti presented a gain-scheduling design technique that relies upon neural models to approximate plant behaviour. J.-E. Zhang showed that every subnetwork of a class of coupled fractional-order neural networks consisting of N identical subnetworks can have $(r + 1)n$ locally Mittag-Leer stable equilibria. Z. Mao and F. Zhao studied the effect of different dimensionless plate lengths and damping values on the UMP in the case of constant flow rate.

In the field of transportation, L. Wang et al. constructed a simulation platform of urban traffic integration and proposed a feasible data analysis, learning and parameter calibration method based on RBFNN. S. Chen et al. developed a multiconstrained network intensive vehicle routing algorithm based on an adaptive ant colony algorithm. D. Gong et al. studied the pinning synchronous problem for complex networks with interval delays and proposed a series of useful theories. Y. E. Shao et al. proposed a two-NN based scheme to enhance the accurate identification rate (AIR) for MCCPs by performing dimension reduction on disturbance categories.

On the complex systems of the nature, biology, and the human body, R. Rao and S. Zhong investigated the stochastically exponential stability of reaction-diffusion impulsive stochastic cellular neural networks (CNN). S. Rómoli et al. proposed an online state estimator based on an RBFNN to solve the question of the lack of online information on some bioprocess variables and the presence of modelling and parametric uncertainties. H. Zhou et al. proposed a large-scale neural circuit model and showed that parallel excitatory and inhibitory pathways underlie the learned neural responses across multiple brain regions. Z. Xiao et al. proposed a BP NN for the prediction of aquaculture dissolved oxygen. C. S. Chin et al. used the transfer learning and deep convolutional neural network (CNN) to perform image recognition on the fouling image by classifying the detected fouling species and the density of fouling on the surface.

On the classification, optimization, and diagnostic issues of a complex system, J. Wang et al. used the finite-time stable control method with RBF neural network to suppress the structural vibration. Á. Morera et al. described an experimental study on the suitability of deep NN to three automatic demographic problems. W. Tian et al. presented an optimization method for the design of the layout of an autonomous underwater vehicles (AUV) fleet to minimize

the drag force. Q. Wang et al. proposed semiactive non-smooth control algorithm with deep learning to suppress harmful effect for building the structure by a surface motion. W. Liu et al. used a RBFNN model to analyse the impacts of six equities on the acquisition of university spin-outs (USOs). J. Ma et al. introduced the multikernel function and cost-sensitive mechanism to construct the fault diagnosis model of check valve based on the multikernel cost-sensitive extreme learning machine (MKL-CS-ELM).

The selected papers in the special issue could not cover all the recent advances of the NN technology for complex systems, yet they present the state-of-the-art progress in this area. We hope these valuable papers may enrich the knowledge in the complex systems community and provide guidance to the readers who are interested in this topic.

Acknowledgments

The guest editors would like to acknowledge and appreciate the authors and the reviewers for their contribution towards the success of this special issue.

Chenguang Yang
Jing Na
Guang Li
Yanan Li
Junpei Zhong

Research Article

Visual Semantic Navigation Based on Deep Learning for Indoor Mobile Robots

Li Wang,¹ Lijun Zhao ¹, Guanglei Huo,² Ruifeng Li,¹ Zhenghua Hou,¹ Pan Luo,¹ Zhenye Sun,¹ Ke Wang,¹ and Chenguang Yang ³

¹State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin 150001, China

²HNA Technology Group, Shanghai 200122, China

³Key Laboratory of Autonomous Systems and Networked Control, College of Automation Science and Engineering, South China University of Technology, Guangzhou 510640, China

Correspondence should be addressed to Lijun Zhao; zhaolj@hit.edu.cn

Received 14 July 2017; Accepted 11 February 2018; Published 22 April 2018

Academic Editor: Thierry Floquet

Copyright © 2018 Li Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to improve the environmental perception ability of mobile robots during semantic navigation, a three-layer perception framework based on transfer learning is proposed, including a place recognition model, a rotation region recognition model, and a “side” recognition model. The first model is used to recognize different regions in rooms and corridors, the second one is used to determine where the robot should be rotated, and the third one is used to decide the walking side of corridors or aisles in the room. Furthermore, the “side” recognition model can also correct the motion of robots in real time, according to which accurate arrival to the specific target is guaranteed. Moreover, semantic navigation is accomplished using only one sensor (a camera). Several experiments are conducted in a real indoor environment, demonstrating the effectiveness and robustness of the proposed perception framework.

1. Introduction

Enabling robots to navigate autonomously in a real world environment is a very challenging topic in the field of robotics associated closely with signal processing, machine vision, and so forth. A robot should have adaptive capacities of planning optimal paths in maps when implementing tasks [1]. Traditional navigation approaches strongly rely on metric or topological maps and constraints which are described in terms of geometry, assuming the shortest path to be the best [2–4]. However, human navigation does not depend on the “best,” but on what to be seen [5]. Semantic information can be further abstracted from images to decide where we go based on it. Normally, we can recognize rooms, corridors, doors, aisles, and so on for reference to plan the motion from one place of a room to another in a building. Moreover, we should also know the exact side within the scenario in order to keep moving on the right path. In other words, we can adjust back if we realize that we are walking in a skew direction. Therefore, mobile robots should

have the abilities mentioned above to perform human-like navigation.

Semantic navigation is regarded as a system considering semantic information to express the environment and then to implement the robot’s localization and navigation. In recent decades, a great deal of attempts have been made focusing on finding applicable solutions for robot semantic navigation. Semantic navigation approaches usually adopt topological structures [6–8], in which semantic places and objects are abstracted to different nodes. It is expected that each node is observed accurately during the motion. However, those nodes may not be observed straightforwardly via the motion offset of mobile robots. Moreover, humans’ navigation depends on their two eyes, which is the motivation behind equipping multiple sensors on mobile robots when dealing with the navigation task.

The main contribution of this paper is to propose a three-layer perception framework based on transfer learning using only visual information, including place recognition model, rotation region recognition model, and “side” recognition

model. Using this framework, semantic navigation can be achieved via only one camera and the motion offset of mobile robots can be solved. Different from traditional semantic navigation methods, the proposed algorithm uses transfer learning to train and recognize the semantic information in the environment and only uses one RGB camera to realize the whole semantic mapping and navigation. Through the recognition of input images, it can provide the robot with key semantic information for navigation, such as navigation in corridors and recognition of turning areas.

The rest of this work is organized as follows. After discussing some related work in Section 2.1, Section 2.2 discusses the details of the proposed three-layer perception framework. Section 3 shows some experimental results obtained by our approach. Finally, Section 4 concludes the paper.

2. Materials and Methods

2.1. Related Work. Semantic information has been used to infer the indoor environment information and to improve the planning efficiency [5, 9–11]. Also, it has drawn a deal of attention in the area of large-scale navigation, seeking to deal with problems in a higher dimension [12]. This type of navigation is inspired by humans, where places are not described in terms of a global map but by semantic information. Semantics in mobile robot navigation has been mainly used for place recognition, allowing mobile robots to build relationships based on places [13]. The topological structure is usually adopted for the semantic navigation, which allows robots to plan their paths at a high dimension [14, 15]. In topological structure, places are often abstracted to nodes, and visiting orders are abstracted to edges.

A variety of approaches are attempted to solve the semantic navigation problem in different perspectives; for instance, Joseph et al. [16] used a human motion mode to predict a path based on how real humans ambulated towards a goal while avoiding obstacles. Posada et al. [17] presented a semantic navigation approach which could be parsed directly from natural language (e.g., “enter or get out of the room, follow the corridor until the next door, etc.”). Zhao and Chen [18] encoded scene information, semantic context, and geometric context into a condition random field (CRF) model, which computed a simultaneous labeling of image regions into semantic classes and structural object classes. Horne et al. [19] used semantic labeling techniques to achieve path planning. In these systems, each pixel in images was classified automatically into a semantic class, and then an image was produced from the induced visual percepts that highlighted certain classes. Recently, neural networks based on learning have been widely used in robots [20]. The deep learning method has become a significant way to solve semantic navigation problems showing the powerful ability to obtain semantic information [21–23]. Zhu et al. [24] proposed a target-driven visual navigation method using a reinforcement learning model that generalizes across targets and scenes. Furuta et al. [25] proposed semantic map based navigation which consisted of generating a deep learning enabled semantic map from annotated world and object

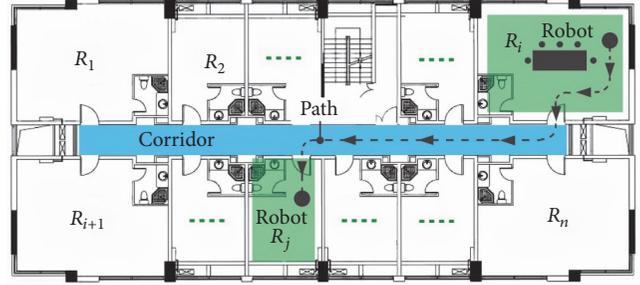


FIGURE 1: The diagram for mobile robot working in an indoor environment. There are several rooms and a corridor in the diagram. A trajectory with a dashed line shows a path for the navigation from a room to another.

based navigation using learned semantic map representation.

Most approaches mentioned above have two main problems:

- (1) Each node in its topological structure is a specific target, which may not be observed through the motion offset of mobile robots on the edge.
- (2) More than one sensor is used, such as a camera for image collection and a laser for mobile robot mapping and motion.

The two problems have motivated our current work, aiming at achieving visual semantic navigation in a human-like way using only one camera.

2.2. Visual Semantic Navigation Based on Deep Learning. People achieve the perception of the environment through images seen by eyes and then guide the behavior. Therefore, we can learn from the “perception-guidance” model to control the robot. In this paper, a three-layer perception framework based on transfer learning is conducted with a common scene (composed of multiple rooms and corridors, as shown in Figure 1). This framework can only rely on image information of a single camera to perceive the surroundings and identify the region where the robot stands and the current pose, which provides decision information for semantic navigation.

2.2.1. Three-Layer Perception Framework. Mobile robots usually work in the environment shown in Figure 1. It can be supposed that the number of rooms is n ($n \in \mathbb{N}^+$) and the semantic task is to move the robot from a room named R_i ($i < n$, $i \in \mathbb{N}^+$) to R_j ($j < n$, $j \in \mathbb{N}^+$). To achieve this semantic task, the robot is required to determine the initial semantic region firstly and then plan the path to reach the target region (the dashed line for the navigation path as shown in Figure 1). As the input information of the robot is merely images acquired by a camera, the learning algorithm can be used to train the robot’s perception model of the environment to realize the semantic navigation purpose.

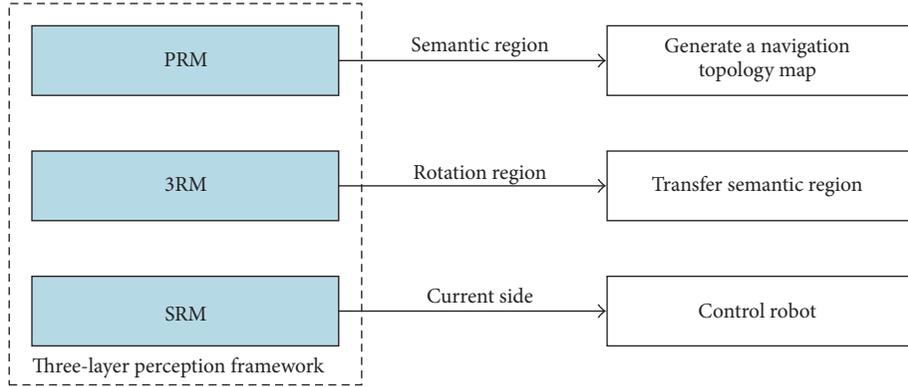


FIGURE 2: Three-layer perception framework.

Each room and each corridor is usually classified as a category, but only the region where the robot is located can be identified. Additional sensors are needed to implement automatic navigation for the robot, although we have already obtained a semantic map. It is difficult to complete the whole semantic navigation in a single neural network model because it cannot provide the robot with all the navigation information simultaneously. Therefore, we design a three-layer perception framework consisting of three perceptual models, which are place recognition model (PRM), rotation region recognition model (3RM), and “side” recognition model (SRM), as shown in Figure 2.

The PRM is used to identify the semantic region where the robot is currently located. And then it gets the navigation topological map according to the semantic task. The 3RM is to identify the key regions when transferring between regions, such as the rotation position at a door when a robot moves from a room to the corridor. The SRM is to provide the relative pose information between a robot and the environment to control the movement. The “side” means that a robot is located at a side. A robot perhaps locates in the left side, center, or right side, when it moves in a corridor or aisle.

(1) *Place Recognition Model (PRM)*. When the robot implements semantic tasks, it is necessary to determine the semantic region where it stands and the target places and then to carry out semantic navigation planning. In an environment similar to Figure 1, the semantic task may be moving from a position of one room to a target region of another room, or from a corridor to a specific region of one room. It is hard to perform navigation planning because the initial position and orientation are both uncertain. There are several main aisles for walking in a room; therefore, several regions can be divided according to these aisles, and each one is regarded as a semantic region. The recognition model of semantic regions can be trained using the method of image classification in machine learning. It needs to collect images of different positions and perspectives in each semantic region as training samples.

Set the number of semantic regions divided in the i th room as n_i , and the number of semantic regions in the corridor

as n_{Corr} ; then, the total number of semantic regions can be calculated by

$$N_{\text{Sem}} = \sum_{i=1}^n n_i + n_{\text{Corr}}. \quad (1)$$

Deep learning is widely used in image classification and has obtained excellent achievements in the ImageNet Challenge; for instance, the top-5 network model accuracy rate of Google’s Inception-V3 reaches up to 96.5% [26, 27]. In addition, transfer learning can use the complex trained neural network model to train the new classification to reduce the amount of training samples and save training time [28, 29]. Therefore, the neural network model for recognizing semantic regions is designed using transfer learning. The Inception-V3 model consists of 11 layers of inception module, which uses multiple branches to extract high-level features of different abstraction levels to enrich the expressive ability. The neural network model framework of semantic region perception based on transfer learning is shown in Figure 3. The model’s input is RGB images. The parameters of Inception-V3 model trained on the ImageNet dataset are used to calculate the network forward transmission, and 2048 nodes are obtained in the bottleneck layer. Then, the last fully connected layer FC is replaced. The number of output categories is the total number of semantic regions N_{Sem} . Then, the Softmax layer is calculated, and the output is the probability of each semantic region category. We use Inception-V3 model for image feature extraction directly and then take the extracted bottleneck feature vector to train a single-layer fully connected neural network.

Suppose that the input RGB image is I , the bottleneck output is y_{bp} ($p \in [1, N_b]$) (the subscript “b” means bottleneck; N_b is the number of bottleneck layer nodes) after calculating function f_{V3} of Inception-V3 module, and the output of FC layer is y_{cq} ($q \in [1, N_{\text{Sem}}]$) (the subscript “c” means fully connected layer).

The bottleneck output can be calculated by

$$y_{bp} = f_{V3}(I). \quad (2)$$

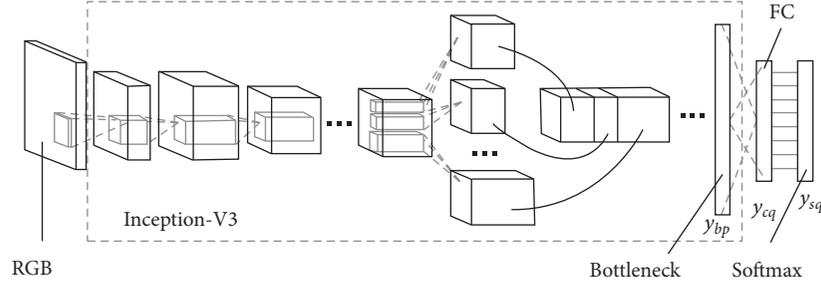


FIGURE 3: The diagram of PRM based on transfer learning.

And the ReLU is selected as an activation function in the FC layer, and then the output of the FC layer can be given by

$$y_{cq} = f_{\text{ReLU}} \left(\sum_{p=1}^{N_b} w_{pq} y_{bp} + b_q \right), \quad (3)$$

where parameters w and b are the weight and bias of the FC layer, respectively.

Model parameters of the FC layer are needed to be trained in the network, and the number of parameters can be given by

$$\text{Num} = N_b \cdot N_{\text{Sem}} + N_{\text{Sem}}, \quad (4)$$

where Num is the amount of parameters.

Finally, the Softmax function is used to obtain the probability of each output. The output y_{sq} ($q \in [1, N_{\text{Sem}}]$) (the subscript “s” means Softmax layer) can be obtained by

$$y_{sq} = f_{\text{Soft max}} (y_{cq}) = \frac{e^{y_{cq}}}{\sum_{q=1}^{N_{\text{Sem}}} e^{y_{cq}}}. \quad (5)$$

For every input image, the probability value belonging to each category can be calculated by the model PRM. And the category with a maximum probability is the final recognition result.

(2) *Rotation Region Recognition Model (3RM)*. Through the place recognition model, the robot can recognize the region location where it stays, but it also needs the recognition information of region transfer to reach the target one, such as how to reach the door from a region of the room. The robot usually performs rotation during the transfer among semantic regions, and it needs to recognize where to rotate. In order to realize the recognition of the rotation region for the robot by the image information, a rotation recognition perception model is proposed based on transfer learning. The rotation position of the robot is identified by sensing room regions and door regions connected with the corridor.

Usually, the movement of the robot between room and corridor is divided into four cases: (a) from room to the left side of corridor, (b) from room to the right side of corridor, (c) from the left side of corridor to room, and (d) from the right side of corridor to room. We can obtain the best rotation region by analyzing the volume of the robot and the turning

radius, which means reaching nearly the center line of the doorway or corridor after rotating. Therefore, the navigation recognition region can be divided at the door as shown in Figure 4.

The navigation recognition regions are drawn as three blue dashed boxes in Figure 4. And RR_{i1} is the recognition region entering the room from the left side of the corridor (Path 1). RR_{i2} is the recognition region entering the room from the right side of the corridor (Path 4). RR_{i3} is the recognition region entering the left or right corridor from the room (Path 2 and Path 3). In brief, RR_{i1} and RR_{i2} are referred to as the entrance recognition regions, and RR_{i3} is called the exit recognition region. The symbol “ i ” in the subscript means the i th room.

The navigation in the room among semantic regions also needs to identify the rotation recognition region, shown in Figure 5. Semantic regions are divided in the room according to the method described in the previous section. The number of semantic regions divided in the i th room is n_i . The robot should recognize the rotating positions between the two adjacent semantic regions. Therefore, it is necessary to determine the location of the recognition region and to collect the images. Recognition regions are described as dashed boxes in Figure 5, and arrows indicate the movement direction of the robot.

In order to determine the number of rotation recognition regions, it is necessary to analyze the distribution of the recognition region in the room. The center line of the region can be connected (such as the red dashed line in Figure 5), and then the required recognition regions can be obtained according to the connection. For connections like the “ T ” type, three regions are required, while the “ L ” type requires two regions.

We can suppose that the number of “ T ” type connections in the i th room is n_T^i , and the number of “ L ” type connections is n_L^i ; then, the number n_{Rot}^i of regions in the i th room can be calculated by

$$n_{\text{Rot}}^i = 3n_T^i + 2n_L^i. \quad (6)$$

In addition, there are three rotation recognition regions at the door of each room; the total number N_{Rot} of regions is given by

$$N_{\text{Rot}} = \sum_{i=1}^n n_{\text{Rot}}^i + 3n. \quad (7)$$

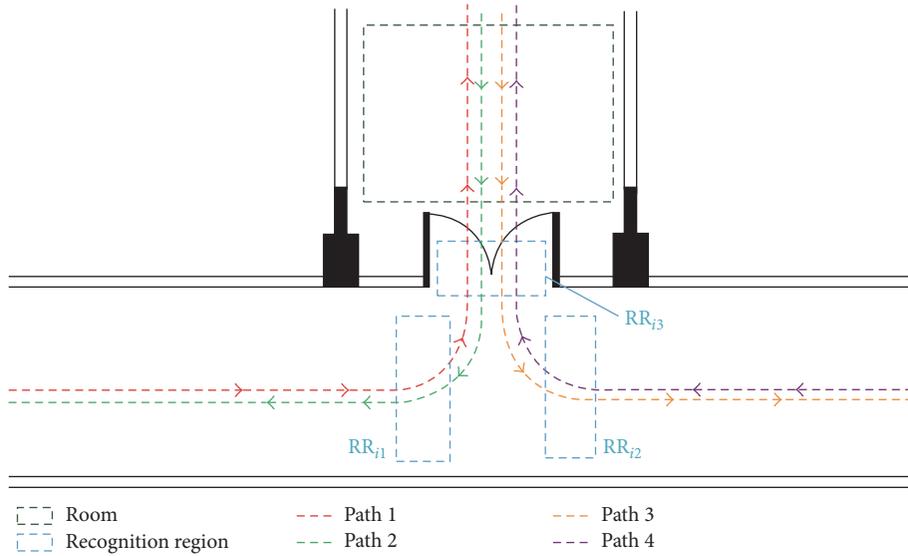


FIGURE 4: The diagram of rotation recognition region at the door. The green dashed box means room area. The three blue dashed boxes mean recognition regions. The four dashed lines show the different movement directions of the robot.

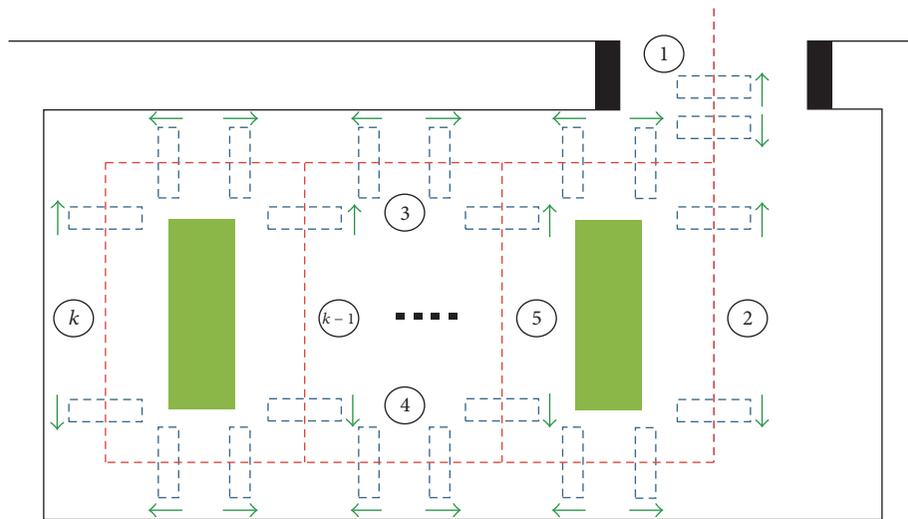


FIGURE 5: Diagram of semantic regions and rotation recognition regions divided in the room. The circled numbers indicate semantic regions. The green blocks describe objects in the room. The robot's accessible area is indicated by the red dashed line. The blue dashed boxes are recognition regions and arrows mean the movement direction of the robot.

When the robot moves between rooms and corridors, it needs to recognize the rotation recognition region firstly and then rotate to the corresponding direction. The locations of recognition regions are determined in each semantic region and images should be collected in the corresponding directions. Then, each region is trained as one category. The location and orientation of the robot are limited in a fixed range for each recognition region when collecting images. In the region RR_{i1} , the robot should be in the center line of the corridor with the direction towards the right. In the region RR_{i2} , the robot locates in the center line of the corridor with the direction towards the left. In the region RR_{i3} , the robot should be at the center of the doorway with the direction towards the corridor (perpendicular to the corridor extension

direction). Besides, we need to collect images outside the rotation recognition region for training in neural networks as nonrecognition region.

The neural network is trained by the method of transfer learning, and the output nodes are the rotation recognition regions and the nonrecognition region; the amount is N'_{Rot} , given by

$$N'_{Rot} = N_{Rot} + 1. \quad (8)$$

As the neural network structure is similar to Figure 2, its structure is not given here. The trained neural network model is used as a rotation recognition region perception model to guide the robot through different semantic regions.

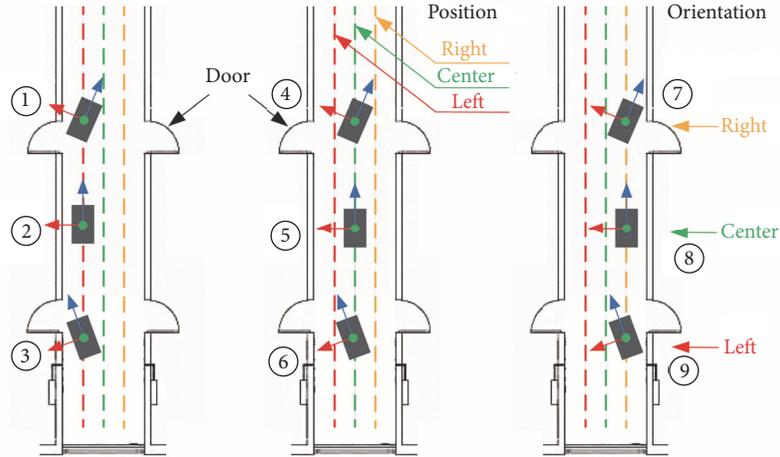


FIGURE 6: Positions and orientations of the mobile robot in the corridor. The nine states of the robot are shown.

(3) “Side” Recognition Model (SRM). In order to reduce the collision when the robot moves in the corridor or rooms, it is necessary to be able to perceive the pose of the robot. As the input for the robot is images, we can learn from the process of people walking in the corridor and then design a “side” recognition model. Firstly, we analyze the operation state of the robot in the corridor. Secondly, the neural network model of recognizing robot pose is trained by transfer learning.

The mobile robot moves in a two-dimensional plane; its pose includes position and orientation. There are nine different states according to the position and orientation when moving in the corridor as shown in Figure 6. Among them, the positions are separated into corridor center, left side, and right side, and the orientations are divided into center direction, left, and right. For convenient description, the nine states of the robot are abbreviated as shown in Table 1. When the robot moves in a room, its pose state is similar to that in the corridor, which has nine states too. As the door region is relatively small, in order to avoid the collision of the robot, it needs to sense and adjust its pose when passing through the door. Thus, the method of image collection at the door of each room is similar to that in the corridor, as shown in Figure 7.

When the robot is in a different state, its camera (fixed on the robot with the forward direction) observes different images. Therefore, we can recognize the pose state through image classification. Similar to the training model above, a neural network model of robot recognition in the corridor is designed using transfer learning. The last full connection layer of Inception-V3 model is modified to output the nine states of the robot, and then the single-layer fully connected neural network is trained.

Images that robots observed in different poses need to be collected when training the model. For facilitating control and reducing the number of perception models, images at the same pose state are put together, as a category to train network model. To cover possible scenarios, data collection takes two movements in the corridor and doorway.

TABLE 1: Nine pose states of mobile robot.

Orientation (O)	Position (P)		
	Right (R)	Center (C)	Left (L)
Right (R)	PROR	PCOR	PLOR
Center (C)	PROC	PCOC	PLOC
Left (L)	PROL	PCOL	PLOL

TABLE 2: Robot poses and control strategies.

Robot poses	Control strategies
① PLOR	Turn left and move to center from left
② PLOC	Move to center from left
③ PLOL	Turn right and move to center from left
④ PCOR	Turn left
⑤ PCOC	Move forward
⑥ PCOL	Turn right
⑦ PROR	Turn left and move to center from right
⑧ PROC	Move to center from right
⑨ PROL	Turn right and move to center from right

The initial position and orientation may be in any cases above, and the movement may deviate from the center direction; it is necessary to adjust the control according to each case. The pose states with corresponding control strategies are given in Table 2. The robot’s state can be recognized through the input image, and corresponding control is conducted, which makes the robot move along the center.

2.2.2. *Semantic Navigation of the Mobile Robot.* When the mobile robot performs semantic navigation between multiple rooms and corridors, it is necessary to determine the topological relations between semantic regions according to prior information. An indoor environment usually contains several rooms and corridors; the topological relationship (shown in Figure 8) between any semantic regions can be given combined with the indoor semantic region division (shown

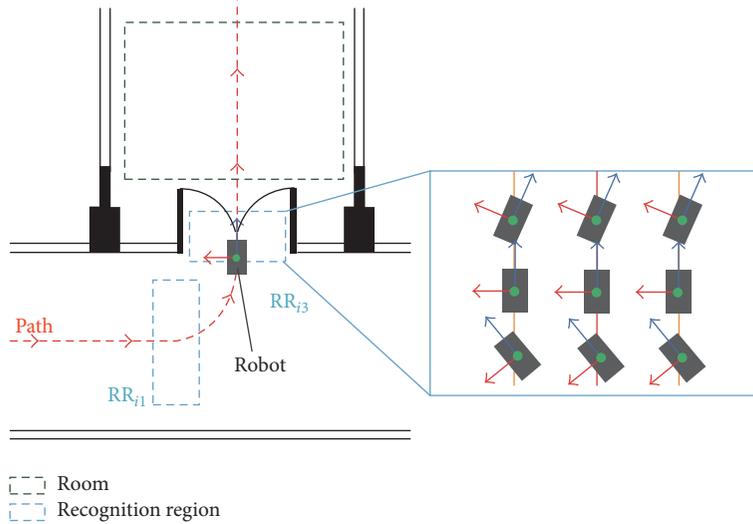


FIGURE 7: Positions and orientations of the mobile robot in the doorway.

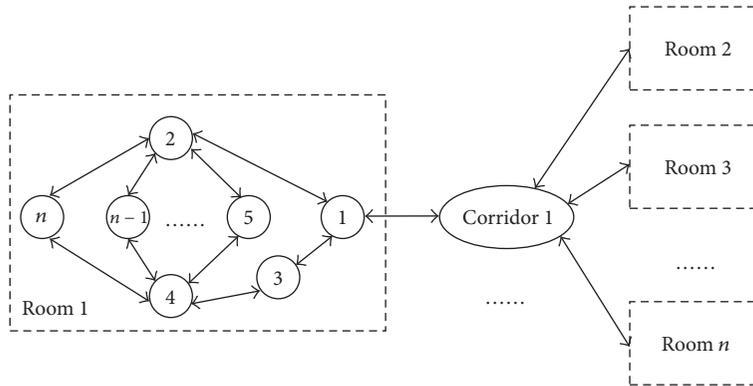


FIGURE 8: Semantic region topology diagram.

in Figure 5). The semantic topological relation diagram is a directed graph of connected nodes, the node is a set of semantic regions, and the edge is a set of rotation recognition regions. The semantic regions are connected by rotation recognition regions. The topological relations between any two semantic regions can be calculated by the directed graph, and the navigation path with the smallest number of semantic regions is selected as the optimal path.

The robot's semantic navigation path can be generated automatically through the topology diagram of Figure 8, which is used to guide the robot motion. Meanwhile, the three-layer perception framework described in Section 3.1 is used to obtain corresponding perceptual information to make decisions. Assuming that the robot is currently in the door region of the room R_i , the semantic task is to reach the room R_j ; then, the robot's decision process is as follows:

- It determines the semantic region using PRM.
- It obtains the pose using the pose perception model and adjusts position and orientation to move towards the door.
- It determines whether the robot is in nonrecognition region or rotation recognition region using 3RM. The

robot keeps going straight if it is in the nonrecognition region. And the robot rotates to corridor when the exit recognition region is detected.

- It obtains the robot's pose relative to the corridor using SRM, and the robot moves along the corridor center line through the control strategy in Table 2.
- The robot rotates towards the room when the entrance recognition region of room R_j is detected using 3RM.
- The robot moves along the center line of the doorway through the control strategies in Table 2.

We achieve the robot semantic navigation from the current region to the target through the algorithm above. In the whole process, the robot only relies on images information for perception and decisions, without using odometer, laser, or other sensor information.

3. Results and Discussion

In order to verify the validity of the proposed visual semantic navigation algorithm, several experiments are carried out on

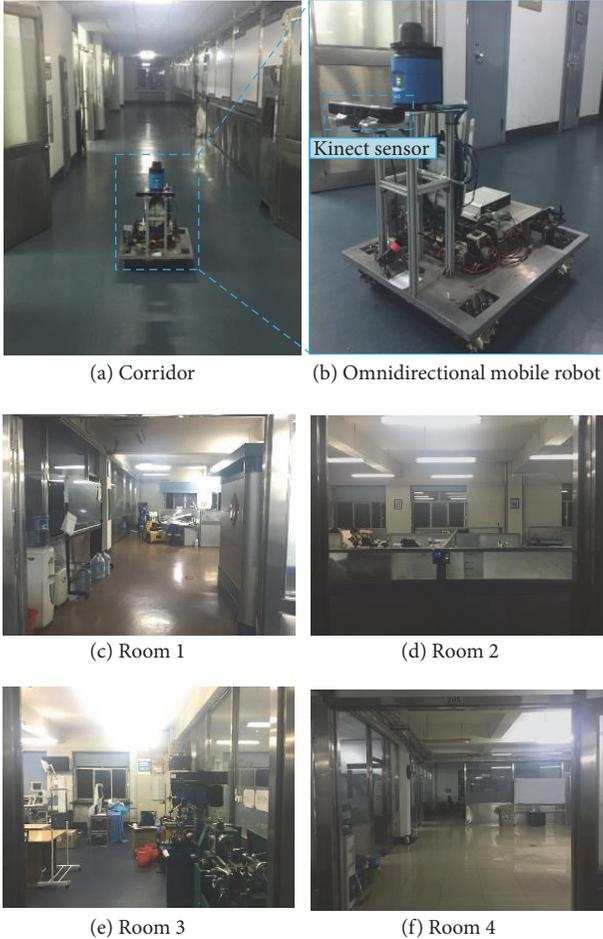


FIGURE 9: Experimental scene and omnidirectional mobile robot.

a mobile robot platform. Firstly, the experimental environment is introduced, and then the training process of the three-level environment perception model is given. Finally, the semantic navigation experiments of the robot from the corridor to room and from room to room are carried out.

3.1. Introduction to the Experimental Environment. An indoor environment including one corridor and four rooms is selected to verify the semantic navigation algorithm, shown in Figure 9. The experimental mobile robot is an omnidirectional platform using Mecanum wheels. It is able to implement movements in any direction. A Kinect sensor is used in the experiment only using RGB color images.

3.2. Training of the Three-Layer Perception Framework. RGB images are collected using a Kinect sensor to train three perception models in the environment as shown in Figure 9.

3.2.1. Training of PRM. The model is used to classify semantic regions in four rooms and a corridor. We control the robot to move and collect images simultaneously. The robot rotates in a circle when it moves one meter forward. The frame rate of Kinect is 30 f/s, which can capture sufficient training data. In

TABLE 3: Training data for three-layer perception framework.

Model	Output nodes	Images quantity	Test set quantity	Accuracy (%)
PRM	9	74600	7460	96.8
3RM	21	83750	8375	94.2
SRM	9	136580	13658	96.5

each room, two main aisles are selected as semantic regions. The nodes of model's output are nine when adding up a corridor region.

3.2.2. Training of 3RM. The model provides rotation positions for robot navigation among semantic regions. Firstly, it is vital to determine the recognition region at the doorway of each room. According to the description in Section 3.1, we collect images in entrance recognition regions, exit recognition regions, and nonrecognition region. In addition, images at recognition regions in each room should be collected. The trained network model has 21 output nodes according to (7) and (8).

3.2.3. Training of SRM. The model provides position and orientation perception information relative to the environment for motion control. RGB images are collected in the method described in Section 3.1 in the corridor. Respectively, nine categories of images in different positions and orientations of the corridor and rooms are collected. The position and orientation are divided into center, left side, and right side separately.

In addition, we need to collect the corridor images in two directions. The correction strategies for the nine poses are consistent whether the robot is in the corridor or rooms. Therefore, images in the same state are trained as one category and the number of output nodes is nine.

The neural network training is carried out using the transfer learning algorithm with Leadtek Quadro K4200 graphics card. 10% of the sample data is randomly selected as validation set and 10% is selected as test set. The training data and results of the three models are shown in Table 3. The accuracy of test results in the test set is quite high, indicating that the trained neural network models have quiet good recognition effects on semantic regions, recognition regions, and robot poses.

3.3. Visual Semantic Navigation Experiments. In order to verify the validity of the proposed three-layer perception framework for the robot semantic navigation task, experiments of the robot from corridor to room and room to room are carried out.

3.3.1. Semantic Navigation Experiments from Corridor to Room. In order to verify the effectiveness and robustness of the model, semantic navigation experiments of all nine initial poses are carried out. The experimental results are shown in Table 4, which shows the initial position images of the robot in corridor, the corridor images observed by

TABLE 4: Semantic navigation experiment from corridor to room.

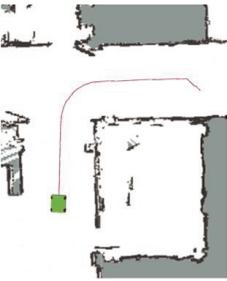
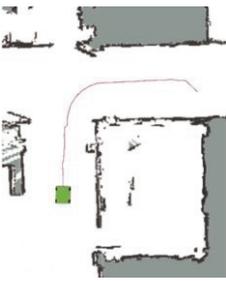
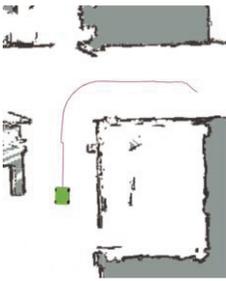
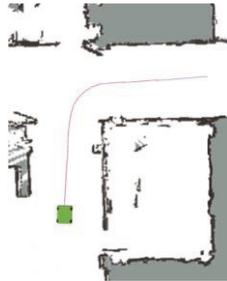
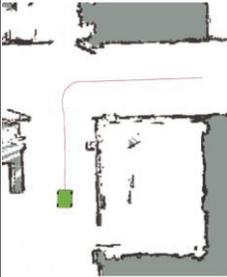
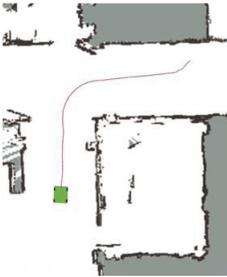
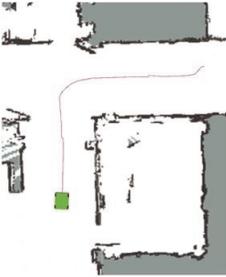
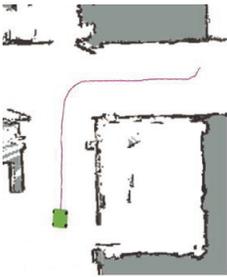
Seq.	Robot states	Initial poses of the robot	Images observed by the robot's camera	Movement trajectories
(1)	PLOR			
(2)	PLOC			
(3)	PLOL			
(4)	PCOR			
(5)	PCOC			

TABLE 4: Continued.

Seq.	Robot states	Initial poses of the robot	Images observed by the robot's camera	Movement trajectories
(6)	PCOL			
(7)	PROR			
(8)	PROC			
(9)	PROL			

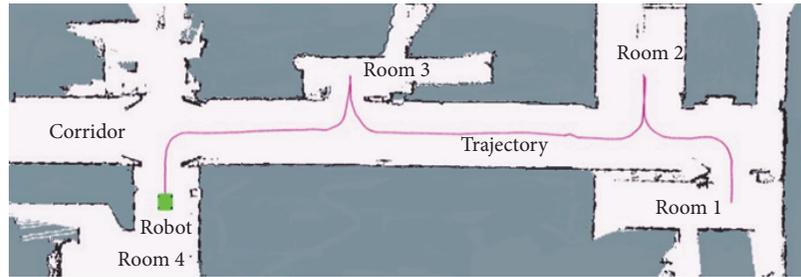


FIGURE 10: Semantic navigation experiment from room to room.

camera, and the trajectories of semantic navigation. It is obvious that the corridor images in different positions are different, which is beneficial to classify. In order to describe the process of semantic navigation, the environment map is established using a two-dimensional laser, and the trajectory of the robot is displayed on the map. From the trajectory diagrams, it can be observed that when there is a deviation between the current pose state and the corridor center, pose adjustment is implemented autonomously. In the doorway, rotation operation is implemented when the robot recognizes the entrance recognition region, and multiple pose adjustments are conducted according to the observed state. The robot can move along the center of the doorway in this method.

The validity and stability of the SRM are verified by the experiments above. The robot can correct the position and orientation by recognizing the current state in different initial poses and correct its own pose in real time to ensure stable movement.

3.3.2. Semantic Navigation Experiments from Room to Room. Semantic navigation experiments from room to room are carried out to verify the validity of the proposed framework. A series of semantic tasks which are “Room 1 → Room 2 → Room 3 → Room 4” are conducted. The robot is initially located at the semantic region of the doorway in Room 1, but the initial information is not given. The semantic tasks are to arrive at regions in the other three rooms, respectively.

The movement trajectory is shown in Figure 10. Firstly, the robot recognizes its semantic region by PRM and generates a semantic navigation topology. Then, the robot rotates to the left into the corridor when the exit recognition region is detected. The robot goes straight until recognizing the entrance recognition region of Room 2 and rotates to the right to enter the room. The same process is implemented to arrive at Rooms 3 and 4. Through the trajectory, it can be seen that the robot can adjust itself to realize the semantic tasks and keep moving along the center line until the target region is recognized.

Experiments show that the three-level perception framework can perform well in the semantic task only using a camera. It provides vital information to guide the navigation. In addition, it can correct the attitude of movement continuously which yields higher reliability and stability.

4. Conclusion

In this paper, a novel visual semantic navigation approach is presented using a three-layer perception framework based on transfer learning. The model comprises place recognition model, rotation region recognition model, and “side” recognition model, which are used to determine the semantic region and recognize the position of the rotating region and the pose information relative to the environment. Only a single camera sensor is employed in our system. Additionally, the “side” recognition model is able to correct the robot’s pose automatically and improve the operational reliability. Semantic navigation experiments are carried out in corridors and rooms, and the results verify the applicability and robustness of our method. We would like to explore the adaptability of changing environment and semantic planning algorithm considering dynamic pedestrians in the future work.

Disclosure

Li Wang and Guanglei Huo are joint first authors.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (61473103, 61673136, and 61473120), the Natural Science Foundation of Heilongjiang Province, China (F2015010), Self-Planned Task (nos. SKLRS201715A, SKLRS201609B, and SKLRS-2017-KF-13) of the State Key Laboratory of Robotics and System (HIT), the Foundation for Innovative Research Groups of the National Natural Science Foundation of China (no. 51521003), and Science and Technology Planning Project of Guangzhou (201607010006).

References

- [1] A. Pandey, S. Kumar, K. K. Pandey, and D. R. Parhi, “Mobile robot navigation in unknown static environments using ANFIS controller,” *Perspectives in Science*, vol. 8, pp. 421–423, 2016.

- [2] H. Omrane, M. S. Masmoudi, and M. Masmoudi, "Fuzzy logic based control for autonomous mobile robot navigation," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 9548482, 10 pages, 2016.
- [3] L. Palmieri, A. Rudenko, and O. A. Kai, "A fast random walk approach to find diverse paths for robot navigation," *IEEE Robotics Automation Letters*, vol. 2, no. 1, pp. 269–276, 2017.
- [4] R. Kümmerle, M. Ruhnke, B. Steder, C. Stachniss, and W. Burgard, "Autonomous robot navigation in highly populated pedestrian zones," *Journal of Field Robotics*, vol. 32, no. 4, pp. 565–589, 2015.
- [5] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, "Robot task planning using semantic maps," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 955–966, 2008.
- [6] D. W. Ko, C. Yi, and I. H. Suh, "Semantic mapping and navigation: A Bayesian approach," in *Proceedings of the 2013 26th IEEE/RSJ International Conference on Intelligent Robots and Systems: New Horizon, IROS 2013*, pp. 2630–2636, Japan, November 2013.
- [7] A. Borkowski, B. Siemiatkowska, and J. Szklarski, "Towards semantic navigation in mobile robotics," *Graph Transformations & Model-driven Engineering-essays*, pp. 719–748, 2010.
- [8] K. Uhl, A. Roennau, and R. Dillmann, *From structure to actions: semantic navigation planning in office environments*, *IROS Workshop on Perception and Navigation for Autonomous Vehicles in Human Environment*, From structure to actions, semantic navigation planning in office environments, 2011.
- [9] T. S. Veiga, P. Miraldo, R. Ventura, and P. U. Lima, "Efficient object search for mobile robots in dynamic environments: Semantic map as an input for the decision maker," in *Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2016*, pp. 2745–2750, Republic of Korea, October 2016.
- [10] D. Pangercic, B. Pitzer, M. Tenorth, and M. Beetz, "Semantic Object Maps for robotic housework - Representation, acquisition and use," in *Proceedings of the 25th IEEE/RSJ International Conference on Robotics and Intelligent Systems, IROS 2012*, pp. 4644–4651, Portugal, October 2012.
- [11] C. Landsiedel, R. De Nijs, K. Kuhnlenz, D. Wollherr, and M. Buss, "Route description interpretation on automatically labeled robot maps," in *Proceedings of the 2013 IEEE International Conference on Robotics and Automation, ICRA 2013*, pp. 2251–2256, Germany, May 2013.
- [12] G. Huo, L. Zhao, K. Wang, and R. Li, "Semantic region estimation of assistant robot for the elderly long-term operation in indoor environment," *China Communications*, vol. 13, no. 5, Article ID 7489969, pp. 1–15, 2016.
- [13] S. Lowry, N. Sunderhauf, P. Newman et al., "Visual Place Recognition: A Survey," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 1–19, 2016.
- [14] R. Drouilly, P. Rives, and B. Morisset, "Semantic representation for navigation in large-scale environments," in *Proceedings of the 2015 IEEE International Conference on Robotics and Automation, ICRA 2015*, pp. 1106–1111, USA, May 2015.
- [15] R. Drouilly, P. Rives, and B. Morisset, "Fast hybrid relocation in large scale metric-topologic-semantic map," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2014*, pp. 1839–1845, USA, September 2014.
- [16] S. L. Joseph, C. Yi, J. Xiao, Y. Tian, and F. Yan, "Visual semantic parameterization - To enhance blind user perception for indoor navigation," in *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo Workshops, ICMEW 2013*, USA, July 2013.
- [17] L. F. Posada, F. Hoffmann, and T. Bertram, "Visual semantic robot navigation in indoor environments," *International Symposium on Robotics*, pp. 1–7, 2014.
- [18] Z. Zhao and X. Chen, "Semantic mapping for object category and structural class," in *Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2014*, pp. 724–729, USA, September 2014.
- [19] L. Horne, J. M. Alvarez, C. McCarthy, and N. Barnes, "Semantic labelling to aid navigation in prosthetic vision," in *Proceedings of the 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 2015*, pp. 3379–3382, Italy, August 2015.
- [20] Y. Jiang, C. Yang, J. Na, G. Li, Y. Li, and J. Zhong, "A brief review of neural networks based learning and control and their applications for robots," *Complexity*, vol. 2017, no. 4, Article ID 1895897, pp. 1–14, 2017.
- [21] J. McCormac, A. Handa, A. Davison, and S. Leutenegger, "SemanticFusion: Dense 3D semantic mapping with convolutional neural networks," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation, ICRA 2017*, pp. 4628–4635, Singapore, June 2017.
- [22] M. Volpi and D. Tuia, "Dense semantic labeling of subdecimeter resolution images with convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 881–893, 2017.
- [23] Y. Xiang and D. Fox, "DA-RNN: Semantic Mapping with Data Associated Recurrent Neural Networks," in *Proceedings of the Robotics: Science and Systems 2017*.
- [24] Y. Zhu, R. Mottaghi, E. Kolve et al., "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation, ICRA 2017*, pp. 3357–3364, Singapore, June 2017.
- [25] Y. Furuta, K. Wada, M. Murooka et al., "Transformable semantic map based navigation using autonomous deep learning object segmentation," in *Proceedings of the 16th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2016*, pp. 614–620, Mexico, November 2016.
- [26] C. Szegedy, W. Liu, Y. Jia et al., "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1–9, Boston, Mass, USA, June 2015.
- [27] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 2818–2826, July 2016.
- [28] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [29] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 1, article no. 9, 2016.

Research Article

Parallel Excitatory and Inhibitory Neural Circuit Pathways Underlie Reward-Based Phasic Neural Responses

Huanyuan Zhou,¹ KongFatt Wong-Lin,² and Da-Hui Wang ^{1,3}

¹*School of Systems Science and National Key Laboratory of Cognitive Neuroscience and Learning, Beijing Normal University, Beijing 100875, China*

²*Intelligent Systems Research Centre, School of Computing, Engineering, and Intelligent Systems, University of Ulster, Magee Campus, Northland Road, Derry BT48 7JL, UK*

³*Beijing Key Laboratory of Brain Imaging and Connectomics, Beijing Normal University, Beijing 100875, China*

Correspondence should be addressed to Da-Hui Wang; wangdh@bnu.edu.cn

Received 13 July 2017; Revised 26 December 2017; Accepted 12 February 2018; Published 12 April 2018

Academic Editor: Junpei Zhong

Copyright © 2018 Huanyuan Zhou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Phasic activity of dopaminergic (DA) neurons in the ventral tegmental area or substantia nigra compacta (VTA/SNc) has been suggested to encode reward-prediction error signal for reinforcement learning. Recent studies have shown that the lateral habenula (LHb) neurons exhibit a similar response, but for nonrewarding or punishment signals. Hence, the transient signaling role of LHb neurons is opposite that of DA neurons and also that of several other brain nuclei such as the border region of the globus pallidus internal segment (GPb) and the rostral medial tegmentum (RMTg). Previous theoretical models have investigated the neural circuit mechanism underlying reward-based phasic activity of DA neurons, but the feasibility of a larger neural circuit model to account for the observed reward-based phasic activity in other brain nuclei such as the LHb has yet to be shown. Here, we propose a large-scale neural circuit model and show that parallel excitatory and inhibitory pathways underlie the learned neural responses across multiple brain regions. Specifically, the model can account for the phasic neural activity observed in the GPb, LHb, RMTg, and VTA/SNc. Based on sensitivity analysis, the model is found to be robust against changes in the overall neural connectivity strength. The model also predicts that striosomes play a key role in the phasic activity of VTA/SNc and LHb neurons by encoding previous and expected rewards. Taken together, our model identifies the important role of parallel neural circuit pathways in accounting for phasic activity across multiple brain areas during reward and punishment processing.

1. Introduction

The ability to adapt to uncertainty is critical for survival and key to wellbeing. To investigate the underlying neural correlates and mechanisms, many experimental and computational studies using stochastic scheduling of reward have been carried out [1–9]. Experimental studies have demonstrated that dopaminergic (DA) neurons in the ventral tegmental area or substantia nigra compacta (VTA/SNc) and neurons in the lateral habenula (LHb) play important roles in encoding uncertainty of reward and punishment [5, 8].

As illustrated schematically in Figure 1 (top row), given some unexpected reward (the presence of an unconditioned stimulus US such as food), DA (LHb) neurons exhibit a phasic

peak (dip) upon the presence of the US [5, 8]. After several trials of learning in the presence of a cue/stimulus, conditioning takes place. The (expected) conditioned cue/stimulus (CS) becomes associated with reward, and the DA (LHb) neurons exhibit a phasic peak (dip) in activity upon the onset of the CS (Figure 1, second row) [5, 8]. Note that the DA and LHb neurons now do not respond to the unconditioned stimulus (US) with a rewarding outcome [5, 8]. One can view this as postreinforcement learning: the agent has learned to completely associate the cue/stimulus CS with the US (e.g., an auditory tone with food), and the latter is no longer needed for further learning. However, if there is an omission of reward (e.g., absence of food), there is an additional dip (peak) in activity for the DA (LHb) neurons (Figure 1, third row) [5, 8].

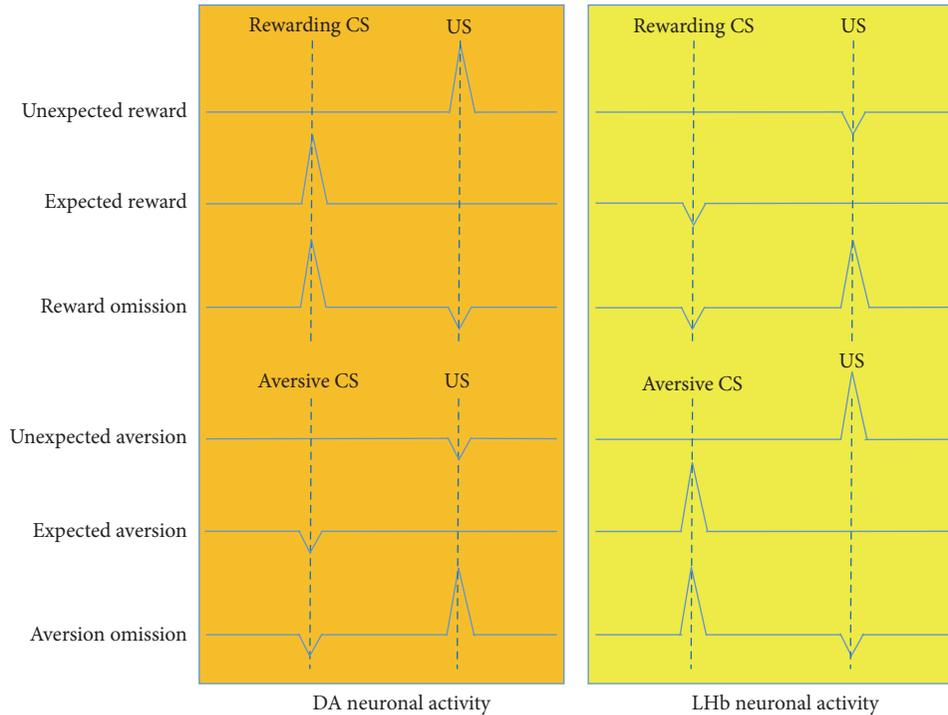


FIGURE 1: Schematic diagram of phasic activity of DA neurons (left orange part) and LHb neurons (right yellow part) given rewarding CS (upper) and nonrewarding/aversive CS (bottom). Each row denotes one situation of outcome.

Instead of the unexpected rewarding outcome US, if we now replace it with an unexpected nonrewarding or aversive stimulus US (e.g., no food or mild electric shock), it has been observed that phasic dip (peak) in the DA (LHb) neurons occurs during the initial phase of the reinforcement learning [5, 8] (Figure 1, fourth row). After learning, this information is transferred to the CS, in which the DA (LHb) neurons exhibit a phasic dip (peak) activity upon CS presentation while staying at baseline activity level during US (Figure 1, fifth row). When there is a sudden unexpected omission of such US or when the US becomes rewarding, then there is a peak (dip) in activity of the DA (LHb) neurons [8, 10, 11] (Figure 1, bottom row). In summary, the phasic activities of DA and LHb neurons signal uncertainty in reward and punishment. Such signaling is also reflected in other brain regions such as the border region of the globus pallidus internal segment (GPb), the internal segment of the globus pallidus (GPi), and the rostral medial tegmentum (RMTg) [2, 3]. However, it is not clear how this information is transmitted within a larger neural circuit.

To understand the underlying computation, previous theoretical and computational studies have applied temporal difference learning [8, 12–15] and neural circuit modeling to understand the phasic activity of DA neurons [16–18] on the basis that the phasic activity of DA neurons acts as a form of reward-prediction error signal [8]. In particular, in the model by Brown et al. [16], there are parallel pathways: one pathway from the cortex through the striosome to VTA/SNc and the other pathway from the cortex through the ventral striatum (VS) to the pedunculopontine nucleus (PPTN)

and VTA/SNc. These two pathways cooperatively control the activity of DA neurons (Figure 2). However, the phasic activity of LHb neurons has not been taken into consideration yet, especially given that LHb has substantial projections to DA neurons in the VTA/SNc [5].

In this work, we propose a large-scale neural circuit model by extending Brown et al.’s [16] model to investigate the phasic activity of not only DA and LHb neurons, but also the extended parts of the network such as the GPb, GPi, and RMTg. In addition to the neural circuit pathways in Brown et al. [16] that control DA signaling (see above), our model also included pathways from the striosome and the VS to the LHb and also one pathway from the LHb to the VTA/SNc via RMTg. These additional pathways are necessary to account for the observed phasic activity of LHb neurons (Figure 2). Further, the pathway from LHb to VTA/SNc via RMTg provides inhibition to the DA neural activity when expected reward was omitted or when there is an aversive outcome. This interareal connectivity is constrained by currently available knowledge from physiological studies (see below for supporting evidence).

Based on simulation results, our model can account for various experimental observations of phasic activation with rewarding or nonrewarding CS, together with or without reward outcomes. Specifically, the model can account for a shift of VTA/SNc and LHb neuron responses from outcome to CS, which agrees with experiments. In addition, the model can also account for the phasic activity of GPb and RMTg neurons, whose responses are similar to those of LHb neurons. Our model shed light on the mechanism of VTA/SNc

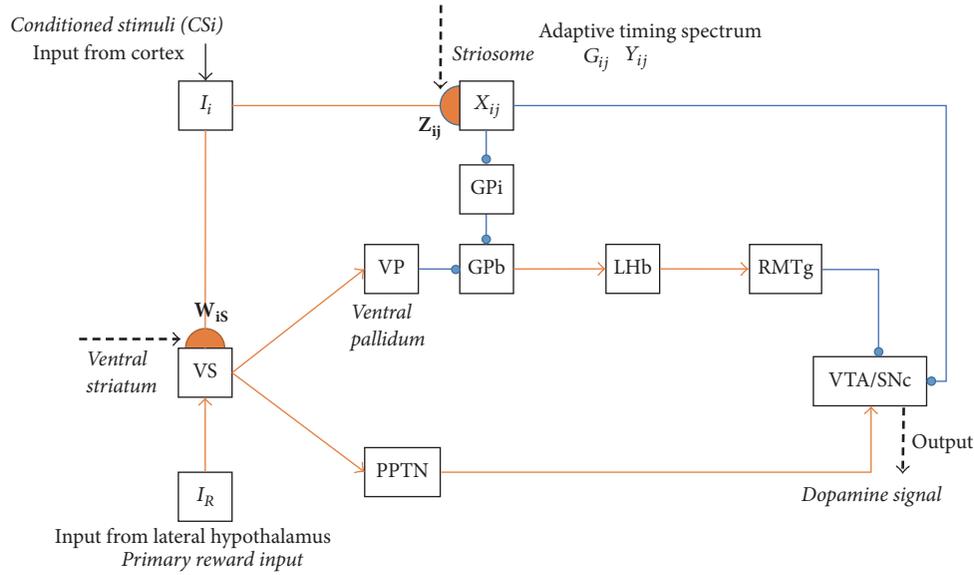


FIGURE 2: Model circuit. Orange arrowheads denote excitatory pathways, blue circles denote inhibitory pathways, and hemidisks denote synapses at which learning occurs. Black dashed lines denote dopaminergic signals. Evidence [21] shows that the ventral striatum (VS) excites PPTN and ventral pallidum (VP). Striosome neurons project to GPi neurons which in turn project to GPb. Dopaminergic (DA) neurons are excited by cortical inputs (I_i) encoding conditioned stimuli and lateral hypothalamus inputs (I_R) encoding unconditioned stimuli via the path VS-VP-GPb-Lhb-RMTg-VTA/SNc and the path VS-PPTN-VTA/SNc path. DA neurons are inhibited by I_i via the path striosome-VTA/SNc. Note that the striosome contains an adaptive spectral timing mechanism and can learn to generate lagged, adaptively timed signals [16]. Lhb neurons are excited by I_i via the path striosome-GPi-GPb-Lhb. Lhb neurons are inhibited by I_i and I_R via the path VS-VP-GPb-Lhb.

and Lhb phasic activity at the neural circuit level, with important roles from the parallel excitatory and inhibitory pathways in the learned responses; namely, (i) the VS-PPTN-VTA/SNc pathway excites DA, while the striosome-VTA/SNc pathway inhibits DA; (ii) the VS-VP-GPb-Lhb pathway inhibits Lhb, while the striosome-GPi-GPb-Lhb pathway excites Lhb; and (iii) the Lhb-RMTg-VTA/SNc pathway magnifies the phasic activity of VTA/SNc. The model is also rather resilient to overall changes in the interregional connections. Finally, our model predicts that the striosome is important since it may remember the timing of the previous reward and provide the comparison signal with the present reward.

2. Materials and Methods

2.1. Model Architecture. Our proposed neural circuit model is schematically shown in Figure 2, which is an extended version of the model proposed by Brown et al. [16]. Namely, we included the GPb, Lhb, and RMTg neural populations into the model based on more recent experimental findings [2, 3, 19, 20]. The details of each part of our model are described as follows.

2.1.1. Lhb Inhibits SNC/VTA via RMTg. Most Lhb neurons are glutamatergic [22], but experiments showed that Lhb inhibits DA neurons. Firstly, *in vivo* recordings demonstrate that most Lhb neurons are excited by a nonreward-predicting cue and are inhibited by a reward-predicting cue when rhesus monkeys perform a visually guided saccade

task [5]. The phasic activity of Lhb neurons is opposite that of DA neurons in terms of responding to outcome valence; Lhb (DA) neurons are excited (inhibited) by non-reward/punishment outcome/cue and inhibited (excited) by reward outcome/cue [5, 8]. Secondly, Lhb neurons respond to cues earlier than DA neurons in unrewarded trials [5]. Thirdly, stimulating Lhb neurons will inhibit DA neurons [21]. The inhibition of Lhb on DA neurons may arise from the direct projection from Lhb neuron to inhibitory interneurons in the VTA/SNc [23] or indirectly through some inhibitory nucleus. In fact, experiments have revealed a path from the Lhb to DA neurons through RMTg and neurons in the RMTg seem to encode aversive stimuli [19, 20]. At the same time, the RMTg transmits negative reward-prediction errors signal of Lhb neuron to positive reward-prediction errors signal of DA neurons [3]. For simplicity, we only include the indirect path from Lhb to DA neurons via GABAergic RMTg.

2.1.2. GPb Excites Lhb. Low intensity electrical stimulation in GPb can evoke a short latency excitatory response in Lhb neurons [21]. The excitation of GPb neurons on Lhb neurons may be mediated by acetylcholine or glutamate [2], or by disinhibition through intra-Lhb interneurons considering the complex microcircuitry within the GP [2, 24]. In addition, glutamatergic projections to Lhb from rat's entopeduncular or primate's GPb neurons have been observed in experiments on nonhuman primates [25, 26]. In brief, there are excitatory projections from GPb to Lhb which form a pathway from GPb to VTA/SNc via Lhb and RMTg [19].

TABLE 1: Model variables.

S	<i>The activation level of ventral striatal neurons</i>
I_i	The i th CS input signal
I_R	The US input signal
W_{iS}	CS-to-VS synaptic weights
G_{WS}	Calcium signal
N^+	Above-baseline dopamine burst signal
N^-	Below-baseline dopamine dip signal
x_{ij}	Striosomal metabotropic response
r_j	Striosomal activity buildup rate parameter
$[G_{ij}Y_{ij}]^+$	Striosomal calcium concentration
Z_{ij}	CS input-to-striosomal synaptic weights
$P_{\text{pre-excite}}$	The level of substance P exciting PPTN
$P_{\text{pre-inhibit}}$	The level of GABA inhibiting PPTN
$VP_{\text{pre-excite}}$	The level of substance P exciting VP
$VP_{\text{pre-inhibit}}$	The level of GABA inhibiting VP
P	The activation level of PPTN neurons
VP	The activation level of VP neurons
$\text{input}_{\text{pre},P}$	The net effect of substance P and GABA on PPTN
$\text{input}_{\text{pre},VP}$	The net effect of substance P and GABA on VP
GPb	The activation level of GPb neurons
LHb	The activation level of LHb neurons
RMTg	The activation level of RMTg neurons
D	The activation level of DA neurons

2.1.3. Conjectured Inputs to GPb from GPI. It has been demonstrated that GPb neurons receive input from the striatum, presumably from the striosome [27]. Hong and Hikosaka [21] have observed that typical neurons in the external and internal segments of the globus pallidus (GPe and GPI) are first inhibited by striatal stimulation but GPb neurons are often (but not always) excited or disinhibited by striatal stimulations. They proposed that signals to GPb should be mediated through inhibitory axon collaterals within the striatum [28] or GPe [24]. Based on these observations, we conjecture that striosome projects to LHb through GPI.

2.1.4. VP Inputs to GPb. In Brown et al.'s [16] model, VP neurons are inhibited by the expectation of reward. However, recent experiments observe that the majority of VP neurons are excited by the expectation of a large reward [21]. Thus, VP-LHb connections could possibly be inhibitory [21]. Therefore, we assume that reward-related signals are transmitted to the LHb through excitatory connections from the GPb and inhibitory connections from the VP.

2.1.5. Excitatory Inputs from VS to VP and PPTN. Although VS neurons are usually identified as GABAergic and inhibit downstream neurons, Hong and Hikosaka [21] showed that the striatal (GABAergic) neurons excite PPTN and VP neurons. The excitation by VS neurons can be mediated by substance P [29, 30]. Thus, we assume that VS directly excites PPTN and VP.

2.2. Dynamical Equations, Input-Output Functions, and Numerical Method. We assume neuronal homogeneity within each brain region, such that each neural population's firing-rate activity within a brain region or nucleus can be dynamically described by ordinary differential equations typically with a decay term plus a term with an input-output function: firing-rate type model (Wilson and Cowan, 1976; see Mathematics and Equations). Specifically, the neural population firing rate (output) is normalized, ranging from zero to one. The input includes constant background input to generate the spontaneous baseline firing activity for each neural population (and brain region) and synaptic terms in the form of coupling strengths to provide the interaction between different neural populations (see Mathematics and Equations). Some of the coupling strengths are subject to change (i.e., plastic) dependent on the presence of reward (see Figure 2). Further modeling details can be obtained from the original model of Brown et al. [16]. The model variables are summarized in Table 1. Parameters are adjusted to fit the observed responses of neurons. Parameter values used for simulations are given in Table 2. In all simulations, numerical integration of the ordinary differential equations was performed with fourth-order Runge-Kutta method [31] using a custom Python code. Codes are available upon request.

2.3. Simulation Protocol. We simulate 200 trials in one block (Figure 3(a)). Every trial lasts for 10 simulated seconds (Figures 3(b)–3(e)). In each trial, we apply different inputs to

TABLE 2: Model parameters.

Symbol	Description	Value
background_{IC}	Baseline of CS input	0.30
background_{IR}	Baseline of US input	0.20
τ	Exponentially decaying time constant of CS/US input	20.0
τ_S	The time constant of VS neurons	36.0
τ_{WS}	The time constant of the change of weight W_{iS}	6
α_{WS}	CS-to-VS weight learning rate	13.0
C_{WS}^{\max}	Maximum CS-to-VS synaptic weight	4.00
β_{WS}	CS-to-VS weight decay rate	13.00
\bar{D}	The baseline activation level of DA neurons	0.194
Γ_D	Phasic dopamine signal threshold	0.001
α_r	Striosomal spectrum spacing	16.5
β_r	Striosomal spectrum offset	30.9
α_G	Calcium activation rate	3.00
B_G	Calcium concentration maximum	5.00
Γ_G	Calcium spike threshold	0.37
β_G	Calcium passive decay rate	12.00
α_Y	Calcium recovery rate	0.108
β_Y	Activity-dependent calcium inactivation rate	48.0
Γ_Y	Calcium inactivation threshold	0.18
α_Z	Striosomal learning rate	500.00
Γ_S	Striosomal output threshold	0.27
A_Z	Maximum CS input-to-striosomal synaptic weight	20.0
B_Z	CS input-to-striosomal synaptic weight decay rate	40.0
τ_{P1}	The time constant of the change of $P_{\text{pre-excite}}$	36.00
τ_{P2}	The time constant of the change of $VP_{\text{pre-inhibit}}$	6.00
W_{SP}	VS-to-pre-PPTN synaptic weight	1.00
τ_{VP1}	The time constant of the change of $VP_{\text{pre-excite}}$	36.00
τ_{VP2}	The time constant of the change of $P_{\text{pre-excite}}$	6.00
W_{SVP}	VS-to-pre-VP synaptic weight	1.00
background_p	The background input to the PPTN	0.10
W_P	VS-to-PPTN input weight	3.00
τ_P	PPTN neurons response time constant	36.00
background_{VP}	The background input to the VP	0.10
W_{VP}	VS-to-VP input weight	3.00
τ_{VP}	VP neurons response time constant	36.00
Γ_{P12}	The difference signal threshold of the excitatory and inhibitory effects previous to PPTN	0.006
Γ_{VP12}	The difference signal threshold of the excitatory and inhibitory effects previous to VP	0.006
τ_{GPb}	GPb neuron response time constant	36.00
background_{GPb}	The background input to the GPb	0.60
W_{VPG}	VP-to-GPb synaptic weight	1.00
W_{SOG}	Striosome-GPb synaptic weight	0.35
τ_{LHb}	LHb neuron response time constant	36.00
background_{LHb}	The background input to the LHb	0.10
W_{GL}	GPb-to-LHb synaptic weight	5.00
Γ_{GPb}	GPb output signal threshold	0.45
τ_{RMTg}	RMTg neuron response time constant	36.00
background_{RMTg}	The background input to the RMTg	0.10
W_{LR}	LHb-to-RMTg synaptic weight	2.00
Γ_{LHb}	LHb output signal threshold	0.25
τ_D	DA neuron response time constant	36.00

TABLE 2: Continued.

Symbol	Description	Value
background_D	The background input to D	0.40
W_{RD}	RMTg-to-VTA/SNc synaptic weight	0.80
W_{PD}	PPTN-to VTA/SNc synaptic weight	1.00
Γ_P	PPTN output signal threshold	0.10
h_D	Maximum hyperpolarization of DA neurons	0.10

	1st~99th trials	100th trial	101st~199th trials	200th trial
<i>Input</i>	Reward CS Reward US	Reward CS Nonreward US	Nonreward CS Nonreward US	Nonreward CS Reward US
<i>Simulate</i>	Learning predictable	Unpredictable	Learning predictable	Unpredictable

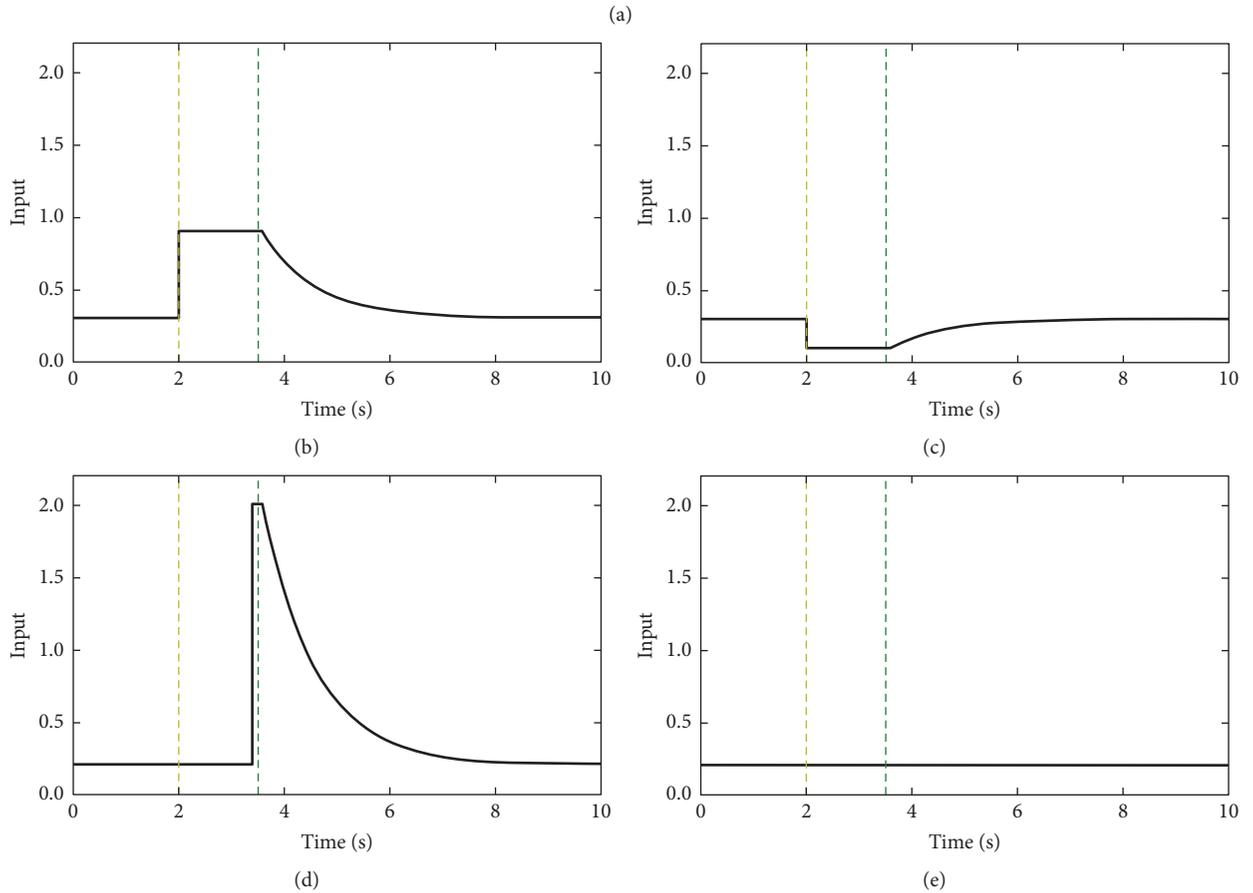


FIGURE 3: Model simulation protocol. (a) Different inputs are applied to simulate different conditions. We simulated a total of 200 trials. In the first 99 trials, we present reward CS input and reward US input to simulate the learning process, which associates the reward CS with the reward US. In the 100th trial, we present reward CS input but nonreward US input; thus, one predicts a reward but does not receive it. In the next 99 trials, we present nonreward CS input and nonreward US input to simulate the learning process, which associates the nonreward US with the nonreward CS. In the 200th trial, we present nonreward CS input but reward US input, simulating the situation where one predicts nonreward but receives it. (b)~(e) Different inputs. The yellow dashed line indicates the time at which CS appears (2.0 s), and the green dashed line indicates the time at which rewards are released or not (3.4 s). (b) Reward CS input. (c) Nonreward CS input. (d) Reward US input. (e) Nonreward US input.

simulate different conditions as follows. First, we simulate the first to the 99th trial with rewarding CS and rewarding US: learning trials. The network can associate the rewarding CS with the rewarding US. The 100th trial is a “test” trial and the network receives rewarding CS and nonrewarding US. We then simulate the unexpected reward condition, that is, nonrewarding CS and rewarding US. From the 101st trial to the 199th trial, the network receives nonrewarding CS and nonrewarding US. The network associates the nonrewarding CS with the nonrewarding US. At the 200th trial, the network receives nonrewarding CS but rewarding US. See Figure 3(a) for a summary of the learning protocol.

We implement different inputs from the cortex to the VS and striosome based on four conditions: reward CS, nonreward CS, reward US, and nonreward US. The rewarding/nonrewarding CS and US are shown in Figure 3 and their mathematical expressions are given in the Mathematics and Equations. Note that the inputs from the cortex are always larger than zero (firing-rate activity cannot be negative in value).

The motivation for such an implementation is based on some observed lines of evidence. First, neurons in the orbitofrontal cortex fire most strongly for cues that predict large reward (with small penalty) and least strongly for cues that predict large penalty (with small reward) relative to neutral conditions (small reward and small penalty) [32, 33]. Second, cortical neurons, including the frontal cortex, are known to exhibit flexibility and mixed response properties; that is, different cortical neurons could have different responses to identical stimuli [34, 35]. For instance, an identical tone could result in different responses from different cortical neurons which could in turn separately transmit information to the same neural “downstream” (e.g., in the midbrain). Third, the expectation values of cue signaling are stored in the cortex but not in the basal ganglia or LHB [36, 37]. The phasic activity of DA neurons can result in plasticity in the cortex and change the representation of cue signaling [38]. In fact, the activity profiles in Figures 3(d) and 3(e) look similar to that of DA release or nonrelease (as measured, e.g., in voltammetry [39]). Also, the sustained or persistent activity in Figure 3(b) could represent (working) memory of the cue, a commonly observed phenomenon in the frontal cortical neurons [36, 37, 40], while the suppressed activity in Figure 3(c) can be thought of as some inhibitory effect with respect to the response in Figure 3(b).

3. Results

3.1. Shift of Phasic Response from US to CS. Many experimental and theoretical studies have reported the shift of DA neurons response from US to CS [41–43]. As discussed previously, in the initial phase of learning, DA neurons are phasically activated from the baseline upon the presentation of an unpredicted reward. An accompanying cue is associated with the rewarding outcome through a learning process. After learning, the phasic activity at reward outcome subsequently decreases to baseline, while a phasic activity now appears upon cue onset (Figure 1).

Our simulation can replicate this trend (Figure 4). When the network receives the rewarding CS and rewarding US (during the first 99 trials), DA neurons exhibit phasic activity upon the US in the first trial (Figure 4(a)). In the second and the subsequent trials, the peak appears upon the CS onset and the previous peak activity upon US onset disappears (Figures 4(b) and 4(c)).

The parallel pathways in our model can account for the shift in neural response from US to CS. At the beginning of the learning phase, CS-to-VS synaptic weights W_{is} and CS input-to-striosomal synaptic weights Z_{ij} are very small or near zero. Thus, the activity of the striosome is maintained at baseline level but the activity of VS has a peak upon US onset. The peak activity of VS then propagates to the LHB through the VS-VP-GPb-LHB pathway, which results in a dip of the LHB activity upon US. Meanwhile, a phasic input to DA neurons through the VS-VP-GPb-LHB-RMTg-VTA/SNc pathway and VS-PPTN-VTA/SNc pathway leads to a phasic activity of DA neurons upon reward US. The phasic activity of DA neurons upon reward US in turn enhances the positive reinforcement-learning signal N^+ (see (7)) which leads to stronger synaptic strengths of afferent inputs to VS and striosome from the cortex: the increased synapse W_{is} and Z_{ij} will enhance CS signal pathways from VS to DA via the PPTN (VS-PPTN-VTA/SNc) and VP (VS-VP-GPb-LHB-RMTg-VTA/SNc), the pathway from striosome to DA (striosome-VTA/SNc), and the pathway from striosome to DA via GPb (striosome-GPi-GPb-LHB-RMTg-VTA/SNc).

The striosome in the model has an adaptive timing spectrum, encoding the timing and the amount of reward associated with the CS [16, 44, 45] (see (10)–(14)). Therefore, through the VS-PPTN-VTA/SNc pathway, rewarding CS can trigger phasic activity of DA neurons (Figures 4(a)–4(c)), while nonrewarding CS can trigger a dip in activity (Figures 5(c)–5(d)). The signal of rewarding US through the striosome inhibits DA neurons at the time when the rewarding US is expected to be present, but the excitation of reward US through the VS to VTA/SNc pathway via PPTN cancels the inhibition of the CS, leading to a baseline activity of DA neurons to reward US (Figures 4(c) and 5(a)). On the contrary, nonrewarding US cannot trigger enough excitation to cancel the inhibition caused by CS in DA neurons, leading to a dip in activity upon nonrewarding US onset (Figure 5(b)).

Experimental studies have shown that the phasic activity of LHB is opposite that of DA neurons in terms of response to reward valence, but with a similar shift in activity to DA phasic activity. In our model, LHB neurons are inhibited and show a dip in their activity upon rewarding US onset (Figure 4(d)). The dip of LHB neural activity shifts from US to rewarding CS in the following and subsequent trials (Figures 4(e)–4(f)). As mentioned previously, unexpected rewarding US can switch on the pathways striosome-GPi-GPb-LHB and VS-VP-GPb-LHB. However, before they are switched on, the rewarding US will inhibit LHB neurons through the VS-VP-GPb-LHB pathway (Figure 4(d)). Once the striosome-LHB and VS-LHB pathways are switched on, the reward CS will effectively inhibit LHB neurons through the VS-VP-GPb-LHB pathway, leading to a dip at the time of the rewarding CS. But the inhibition caused by the rewarding US will be

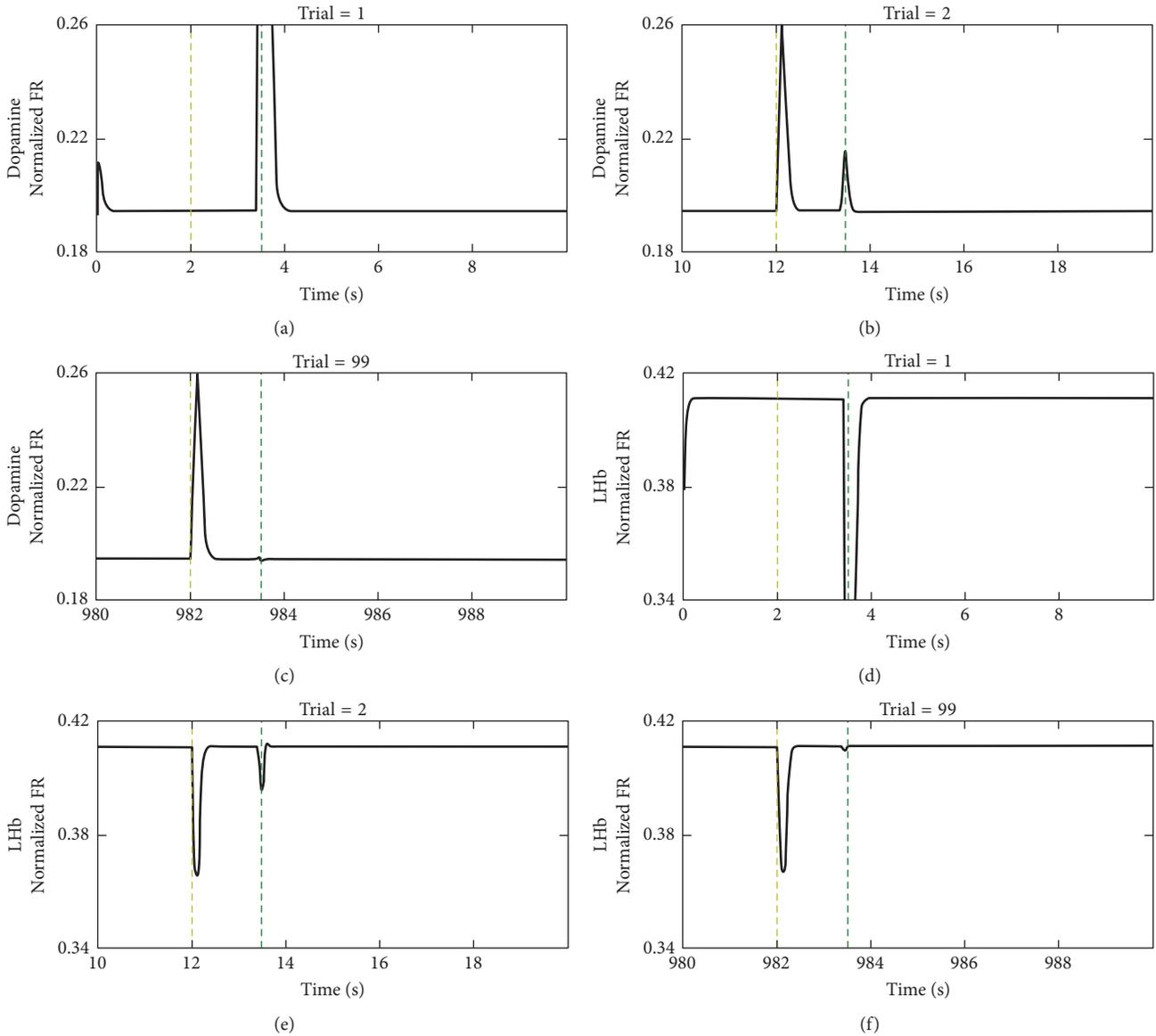


FIGURE 4: The shift of DA and Lhb neurons' responses from US to CS. At the beginning of our simulation, the model circuit receives a reward CS and a reward US. FR: neural firing-rate activity. (a) Response of dopamine neurons in the 1st trial: DA neurons exhibit a phasic peak upon US before learning and do not respond to CS. (b) Response of DA neurons in the 2nd trial: the activity of DA neurons shows a peak upon CS and a peak upon US. The response upon US is weaker than the response in the 1st trial. The responses of DA neurons in the 3rd to 98th trials are similar to (b), but the peak upon US gets weaker over trials. (c) Response of DA neurons in the 99th trial: the activity of DA neurons shows a peak upon CS, but baseline responding to US after learning. (d) Response of Lhb neurons in the 1st trial: Lhb neurons exhibit a phasic dip upon US before learning and do not respond to CS. (e) Response of Lhb neurons in the 2nd trial: the activity of Lhb neurons shows a dip upon CS and a dip upon US. The response upon US is weaker than the response in the 1st trial. The responses of DA neurons in the 3rd to 98th trials are similar to (e), but the dip upon US gets weaker trial by trial. (f) Response of Lhb neurons in the 99th trial: the activity of Lhb neurons shows a dip upon CS, but baseline responding to US after learning. (a), (b), and (c) show the shift of DA neural response from US to CS after learning, while (d), (e), and (f) show the shift of Lhb neural response. The yellow dashed line indicates the time at which CS appears and the green dashed line indicates the time at which rewards are released or not.

canceled by excitation from the striosome-GPi-GPb-Lhb pathway leading to a baseline activity of Lhb neurons at the time of the rewarding US (Figure 4(f)).

3.2. Neural Pathways Underlying Learned Phasic Activity of DA Neurons.

The phasic activity of DA neurons has been

suggested to encode reward-prediction error and to play a pivotal role in reinforcement learning [8, 46, 47]. DA neural activity in our model shows reward-prediction error that is consistent with experimental observations (Figure 5(f)). For instance, after 99 trials of training, the network already can associate the rewarding CS with the rewarding US.

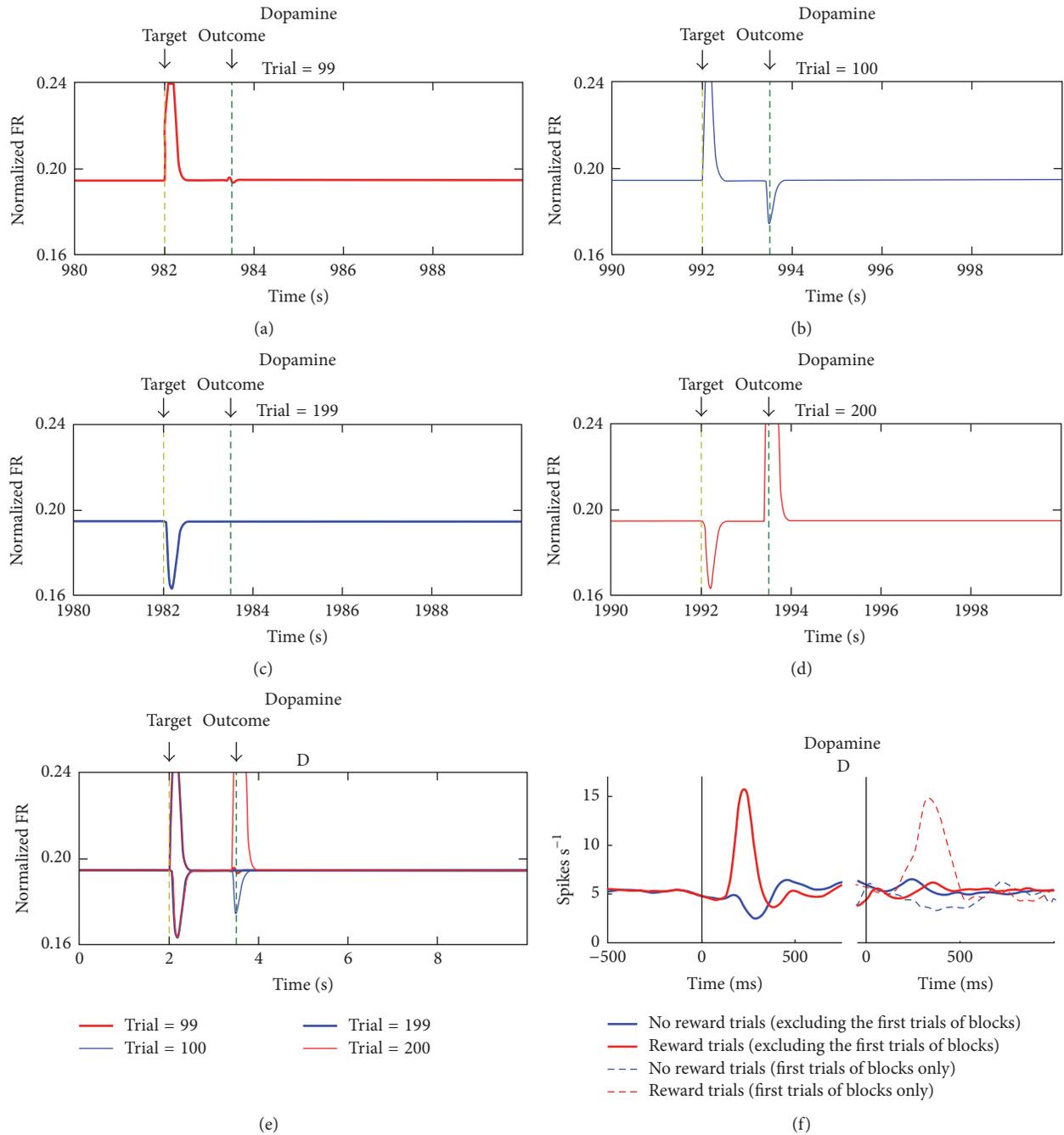


FIGURE 5: Acquired response of DA neurons. (a) The 99th trial: from the 1st to 99th trials, the model circuit receives a rewarding CS and a rewarding US. The result shows that, after learning, DA neurons exhibit a phasic peak upon rewarding CS and a baseline in response to reward outcome. (b) The 100th trial: the model circuit receives rewarding CS and nonrewarding US. The result shows that DA neurons exhibit a phasic peak when rewarding CS appears and exhibit a phasic dip at the time when the reward is expected. (c) The 199th trial: from the 101st to 199th trials, the model circuit receives nonrewarding CS and a nonrewarding US. The result shows that, after learning, the DA neurons exhibit a phasic dip upon nonrewarding CS and a baseline when there is no reward released at this trial. (d) The 200th trial: the model circuit receives nonrewarding CS and rewarding US. The result shows that DA neurons exhibit a phasic dip when nonreward CS appears and exhibit a phasic peak upon reward US. (e) The phasic activity of DA neurons under different situations. The thick red line indicates the activity of DA neurons at the 99th trial, the narrow blue line indicates the activity of DA at the 100th trial, the thick blue line indicates the activity of DA neurons at the 199th trial, and the narrow red line indicates the activity of DA neurons at the 200th trial. The yellow dashed line indicates the time at which CS appears and the green dashed line indicates the time at which rewards are released or not. (f) The physiological experimental result reprinted from Matsumoto and Hikosaka [5]. Red lines indicate reward trials, and blue lines indicate no reward trials. Full lines indicate reward CS-to-reward US (red) and nonreward CS-to-nonreward US (blue), while dashed lines indicate reward CS-to-nonreward US (blue) and nonreward CS-to-reward US (red).

The DA neurons show a phasic activity upon CS onset (at time 2 s in Figure 5(a)). But at the 100th trial, we simulate the condition where the expected reward is omitted. DA neurons are excited right after CS onset (2 s) and inhibited at US presentation (3.6 s) (Figure 5(b)). The network now reassociates the CS with the nonrewarding US after the training from the 101st to 199th trials. The activity of DA neurons then shows a dip at the time when nonrewarding CS is presented at 2 s and shows baseline activity when the nonrewarding US is presented at 3.6 s (Figure 5(c)). Finally, at the 200th trial, we present both the nonrewarding CS and rewarding US to simulate an unexpected reward condition. DA neurons are inhibited upon CS presentation (2 s) but excited at the time when rewarding US is presented once again (3.6 s) (Figure 5(d)). The overall activity profile of DA neurons is summarized in Figure 4(e), which is consistent with experimental observations (Figure 5(f)).

The above phasic responses of DA neural activity associated with the learned stimuli can be understood based on the two parallel pathways in the circuit: the VS-PPTN-VTA/SNc and the striosome-VTA/SNc pathways. It should be noted that, after the 1st trial, the synaptic strengths W_{is} and Z_{ij} are not zero, so VS responds to both rewarding CS and rewarding US. Then, the DA neurons are excited by the rewarding CS through the VS-PPTN-VTA/SNc pathway. When rewarding US is presented, the signal of rewarding CS triggers the activity of striosomal neurons and directly inhibits DA neurons. However, this inhibition is canceled out by the excitation from rewarding US through the VS-PPTN-VTA/SNc pathway. Thus, the activity of DA neurons is effectively maintained at baseline (Figure 5(a)). By the 99th trial, the network has already associated the rewarding CS with rewarding US.

Now, if the rewarding US is omitted (at the 100th trial), no excitation counterbalances the direct inhibition from the striosome, leading to a dip in the activity of DA neurons (Figure 5(b)). This continues until the 199th trial. When the network is presented with a nonrewarding CS followed by nonrewarding US, the direct inhibitory pathway from striosome to DA neurons has been turned off, DA neurons show phasic activity upon nonrewarding CS onset, and the activity of DA neurons is maintained at baseline at the time of nonrewarding US (Figure 5(c)). With a subsequently unexpected rewarding US in trial 200, DA neurons are now excited through the VS-PPTN-VTA/SNc pathway while the nonrewarding CS still causes a dip in the activity (Figure 5(d)).

3.3. Neural Pathways Underlying Learned Phasic Activity of LHb Neurons. Experimental studies have shown that phasic activity of LHb behaves in an opposite way to that of DA neurons [5]. Hence, it has been suggested that LHb neurons play a key role in the coding of the aversive/negative signals [48, 49]. Experiments have been carried out to investigate the activity of several brain nuclei, such as GPb [2] and RMTg [3], to explore the possible functional relationship with these brain regions.

Here, we simulate the activity of these nuclei and the results are consistent with the experimental observations. Our simulations show that the phasic responses of LHb

neurons shift from US to CS. LHb neurons show a phasic dip when the unexpected rewarding US was presented in the first trial (Figure 4(d)). In the following trials, the dip shifts to the time when the rewarding CS presented (Figures 4(e)-4(f)) and baseline activity is observed with rewarding CS (Figure 6(a)) and a small phasic activity upon nonrewarding US (Figure 6(b)). After the training of nonrewarding CS from the 101st to the 199th trials, LHb neurons show a phasic activity upon nonrewarding CS (2 s) while maintaining a baseline level at the time of the nonrewarding US (Figure 6(c)). At the 200th trial, LHb neurons show a peak activity with the nonrewarding CS but a big dip in activity given an unexpected rewarding US (Figure 6(d)). The overall activity profile of LHb neurons (Figure 6(e)) agrees with the experimental observations (Figure 6(f)).

The above-mentioned learned phasic activity of LHb neurons can be explained with two parallel pathways: striosome-to-LHb pathway via GPi and GPb and the VS-to-LHb pathway via VP and GPb. For instance, at the 99th trial, the synaptic strengths W_{is} and Z_{ij} are not zero, which means that the network has already completely associated the rewarding CS with rewarding US. The rewarding CS can inhibit LHb neurons through the inhibitory striatum-VP-GPb-LHb pathway. When the rewarding US appears, the inhibition through the striatum-VP-GPb-LHb pathway will be canceled out by the excitation from the striosome-GPi-GPb-LHb pathway, resulting in a baseline level of LHb neural activity upon reward omission. At the 100th trial, LHb neurons show a dip in the presence of the rewarding CS. But the omission of reward implies that the excitation through striosome-GPb-LHb pathway cannot be canceled out, which leads to a small phasic activity of LHb neurons upon reward omission. At the same time, the synaptic strength Z_{ij} from the cortex to the striosome decreases to zero. When next the nonrewarding CS is paired with a nonrewarding US (from the 101st to the 200th trial), LHb neurons show a phasic activity at the time of the nonrewarding CS onset because of the inhibition through the striatum-VP-GPb-LHb pathway. In the 200th trial, unexpected rewarding signal switches on the inhibitory pathway striosome-GPb-LHb, which leads to a dip in activity of the LHb neurons.

3.4. Learned Phasic Activity of GPb and RMTg. Experiments have shown that the GPb and RMTg neurons display phasic responses to CS and US. In our model, the interaction between striosome-GPi-GPb pathway and VS-VP-GPb pathway leads to the phasic activity of GPb neurons upon CS and US presentation. In particular, the GPb, LHb, and RMTg are also connected with effectively excitatory synapses (Figure 2), and hence their phasic activities should be correlated with that of the LHb, with the same explanations of activity profiles as for the LHb (Figures 7 and 8). Moreover, the LHb-RMTg-VTA/SNc pathway only magnifies the phasic activity of DA neurons and does not qualitatively change the activity profile of DA neurons.

3.5. Robustness Analysis of Two Parallel Pathways' Model. Having shown the important role of the parallel circuit pathways in reproducing the phasic activities observed in

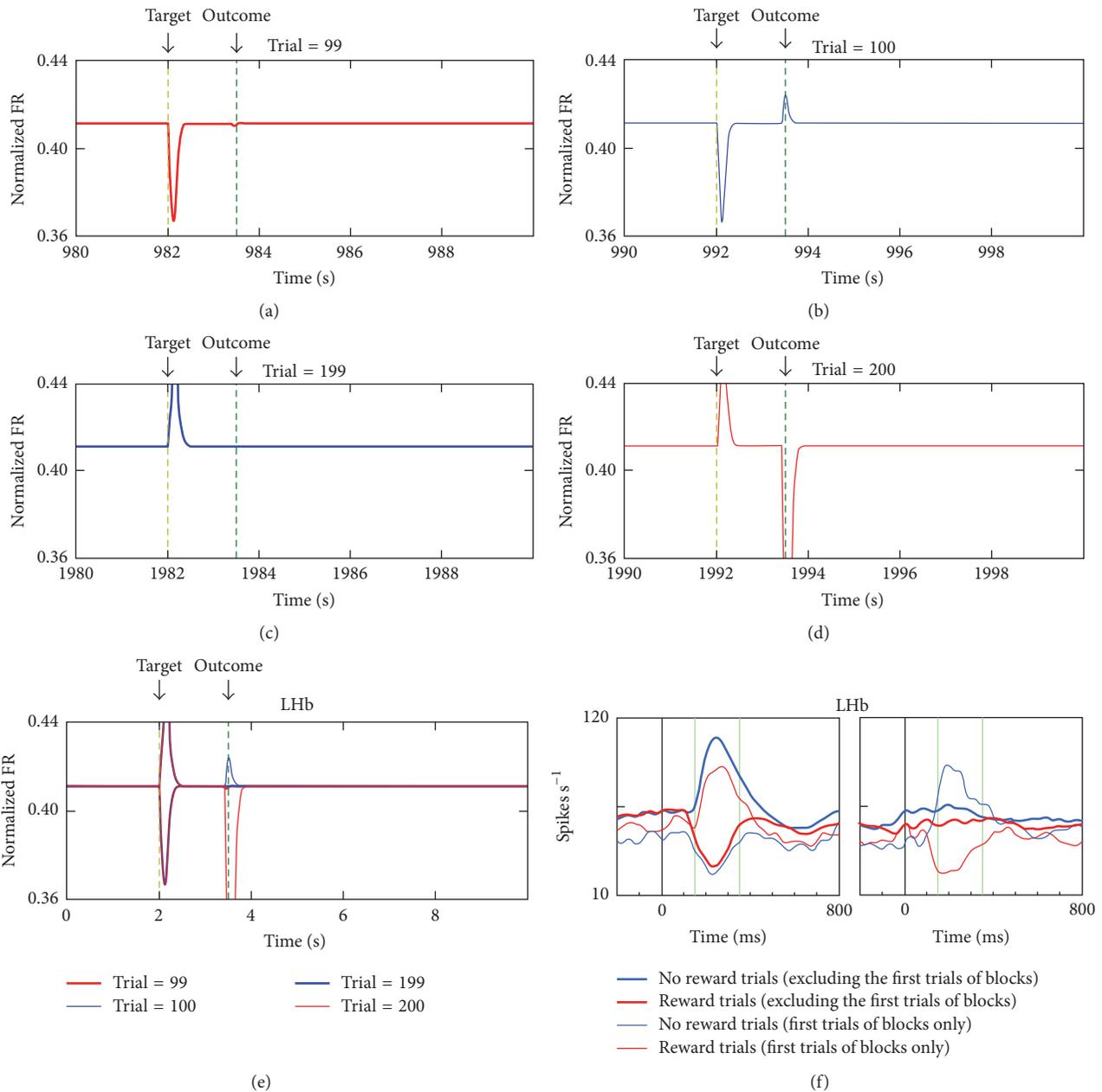


FIGURE 6: Acquired response of LHB neurons. (a) The 99th trial: from the 1st trial to the 99th trial, the model circuit receives rewarding CS and rewarding US. The result shows that, after learning, LHB neurons exhibit a phasic dip upon rewarding CS and a baseline activity in response to rewarding outcome. (b) The 100th trial: the model circuit receives rewarding CS and nonrewarding US. The result shows that LHB neurons exhibit a phasic dip when rewarding CS appears and exhibit a phasic peak at the time when the reward should be released. (c) The 199th trial: from the 101st trial to the 199th trial, the model circuit receives nonrewarding CS and nonrewarding US. The result shows that, after learning, LHB neurons exhibit a phasic peak upon nonrewarding CS and a baseline activity due to omission of reward at this trial. (d) The 200th trial: the model circuit receives nonrewarding CS and rewarding US. The result shows that LHB neurons exhibit a phasic peak when nonrewarding CS appears and exhibit a phasic dip upon rewarding US. (e) The phasic activity of LHB neurons under different situations. The thick red line indicates the activity of LHB at the 99th trial, the narrow blue line indicates the activity of LHB at the 100th trial, the thick blue line indicates the activity of LHB at the 199th trial, and the narrow red line indicates the activities of LHB at the 200th trial. The yellow dashed line indicates the time at which CS appears and the green dashed line indicates the time at which rewards are released or not. (f) The physiological experimental results reprinted from Hong and Hikosaka [2]. Red lines indicate reward trials, and blue lines indicate no reward trials. Thick lines indicate reward CS-to-reward US (red) and nonreward CS-to-nonreward US (blue), while narrow lines indicate reward CS-to-nonreward US (blue) and nonreward CS-to-reward US (red).

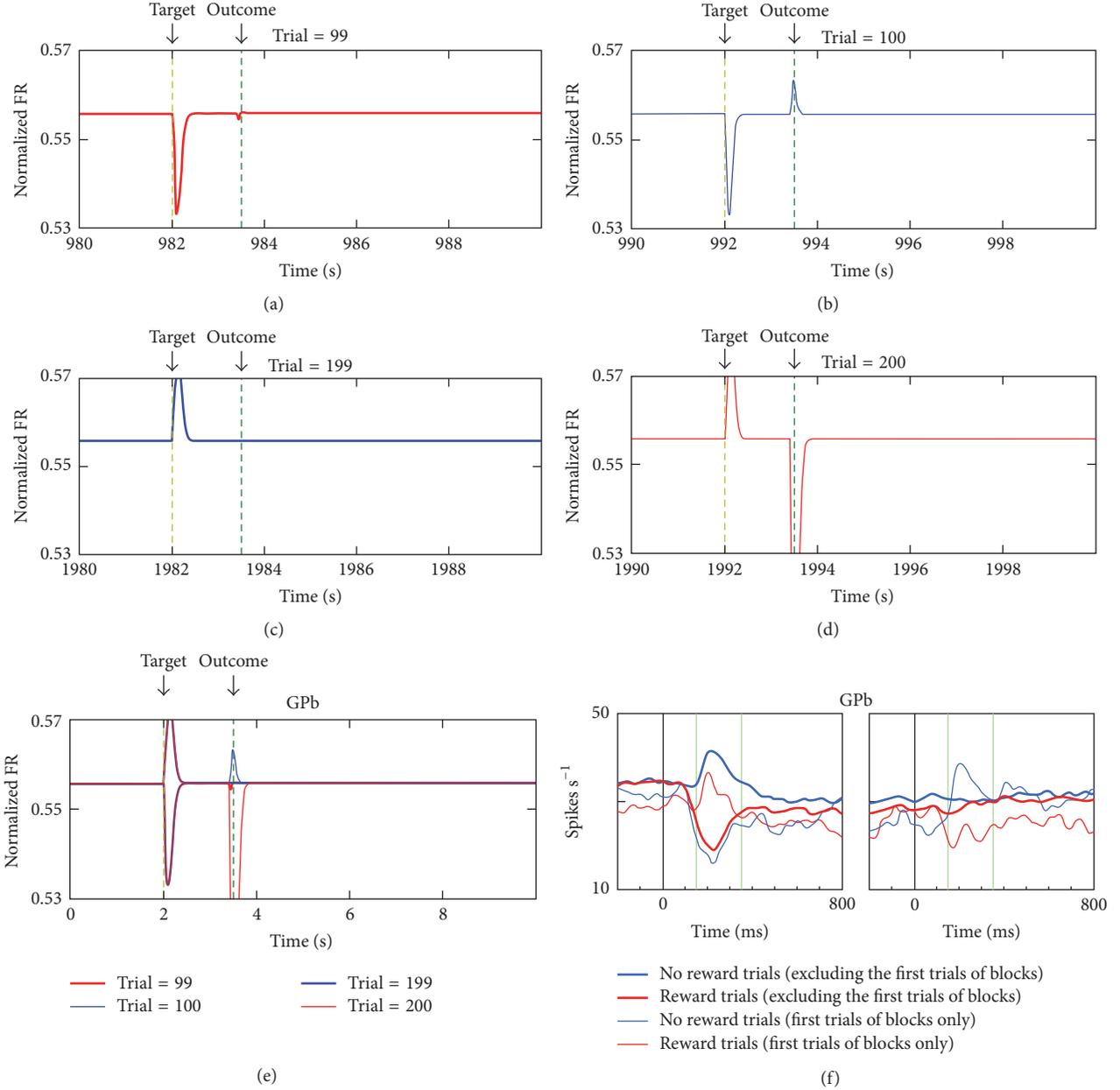


FIGURE 7: Acquired response of GPb neurons. (a)~(e) Similar to Figure 6. (f) The physiological experimental result reprinted from Hong and Hikosaka [2].

experiments, we next further investigate the robustness of the phasic activities in our model with respect to connectivity strength variation. Specifically, we increase or decrease all synaptic weights by 10% and monitor how the phasic activities change.

First, we found that the phasic activities of DA and Lhb neurons did not change substantially when we increased or decreased the following synaptic weights by 10%: W_{SVP} , W_{RS} , W_{SP} , W_{PD} , W_{SOG} , A_Z , and C_{WS}^{\max} (data not shown). Second, weights of synapses on the pathway VP-GPb-Lhb-RMTg-VTA/SNc were found to influence the tonic baseline activity of DA neurons, which we define as \bar{D} . Hence, we change \bar{D} while maintaining the phasic activity of DA and Lhb neurons

when we increase or decrease the weights of the synapses along this pathway (see Table 3). In Figures 9 and 10, we show the activity of DA neurons and Lhb neurons given three different sets of synaptic weights from VP to GPb and corresponding baseline activities \bar{D} . We can see that DA and Lhb neurons continue to demonstrate their characteristic phasic activity profiles. In brief, our neural circuit model is robust to the variation of synaptic weights.

4. Discussion

We extended a previous neural circuit model [16] by incorporating the nuclei GPb, Lhb, and RMTg, and the model

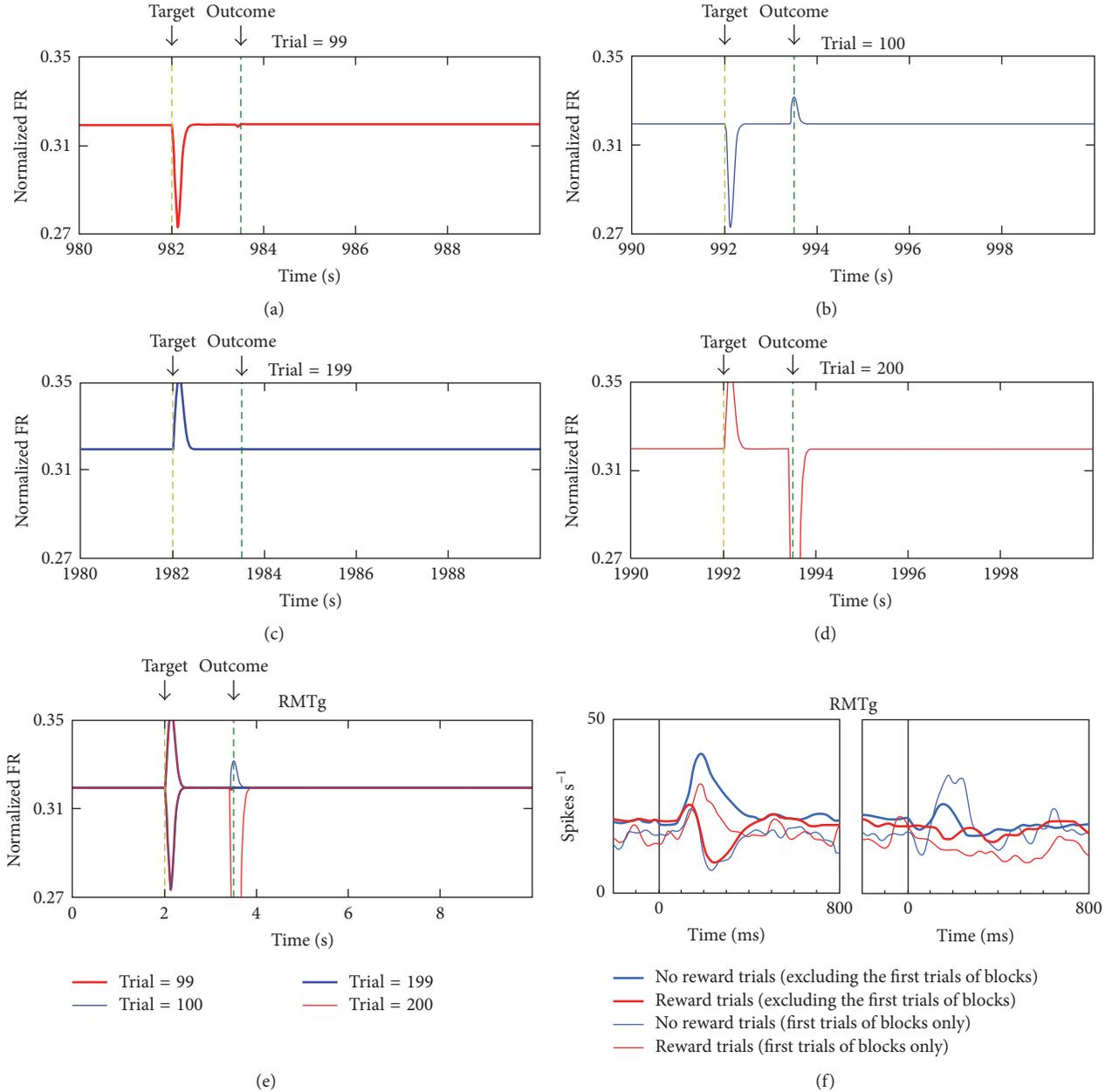


FIGURE 8: Acquired response of RMTg neurons. (a)~(e) Similar to Figure 6. (f) The physiological experimental result from Hong et al. [3].

could account for various experimental data from separate works. Specifically, the model could exhibit the shift of DA and LHb neural responses from US to CS presentation times. Our simulations also replicated the phasic activity of DA, LHb, GPb, and RMTg neurons observed in experiments. The DA (LHb) neurons exhibited a phasic peak (dip) upon reward CS and maintenance of baseline activity in response to a rewarding outcome but a phasic dip (peak) if the reward is omitted. By contrast, the DA (LHb) neurons exhibited a phasic dip (peak) in response to a nonrewarding CS or punishment CS and maintenance of baseline activity in response to the nonrewarding US, but a phasic peak (dip) if a reward occurs or the aversive US is omitted. The acquired

TABLE 3: Baseline activity of DA neurons given increased or decreased synaptic weights.

Synaptic weight	+10%	-10%
W_{VPG}	0.20307 (4.508%)	0.18608 (-4.235%)
W_{GL}	0.17691 (-8.955%)	0.21327 (9.758%)
W_{LR}	0.18006 (-7.334%)	0.20875 (7.431%)
W_{RD}	0.16571 (-14.719%)	0.22102 (13.746%)

responses of GPb and RMTg neurons are similar to that of LHb neurons. These acquired responses are consistent with experimental data [2, 3, 5, 8] and behavioral experiments [50].

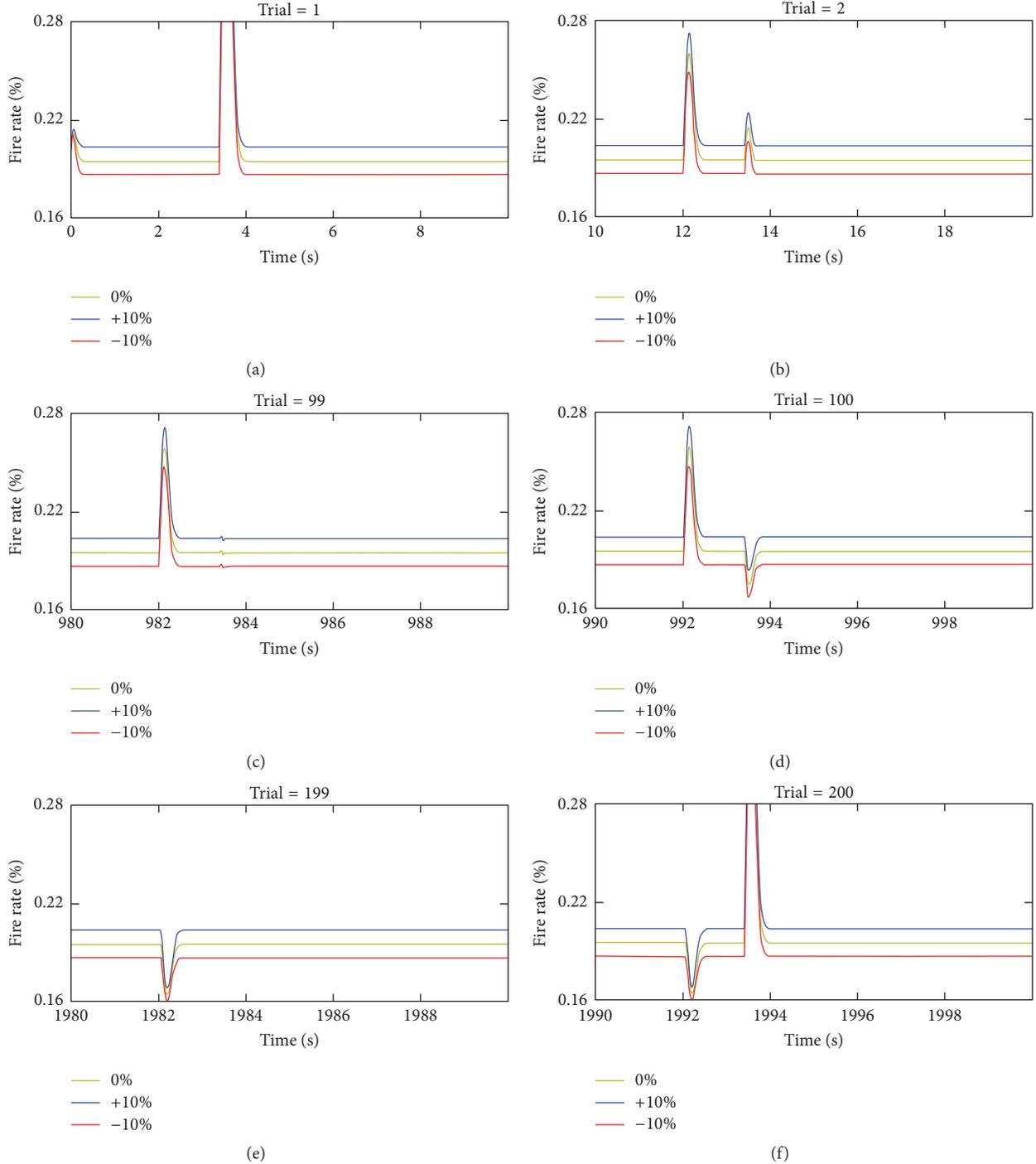


FIGURE 9: The phasic activity of DA neurons given different weights of synapses from VP to GPb. Yellow lines indicate the activity of DA neurons when W_{VPG} equals 1.00 and \bar{D} equals 0.19431, blue lines indicate the activity when W_{VPG} equals 1.10 and \bar{D} equals 0.20307, and red lines indicate the activity when W_{VPG} equals 0.90 and \bar{D} equals 0.18608. (a) Trial 1: phasic peak activity responds to unconditional reward. (b) Trial 2: the phasic activity shifts to the cue. (c) Trial 99: the phasic activity upon the cue and baseline activity upon the reward. (d) Trial 100: the dip activity upon reward omission. (e) Trial 199: the dip activity upon nonrewarding cue. (f) Trial 200: the peak activity upon unexpected reward.

Our model provides insights into the neural circuit mechanism of DA and LHb phasic activity. In particular, parallel excitatory and inhibitory pathways underlie the learned responses: striatum-to-PPTN-to-VTA/SNc pathway excites DA, while striosome-VTA/SNc pathway inhibits DA;

striatum-to-VP-to-GPb-to-LHb pathway inhibits LHb, while striosome-to-GPb-to-LHb pathway excites LHb; LHb-to-RMTg-to-VTA/SNc pathway magnifies the phasic activity of DA. Under different task conditions, we apply different CS and US inputs to the model. The model has a feedback loop

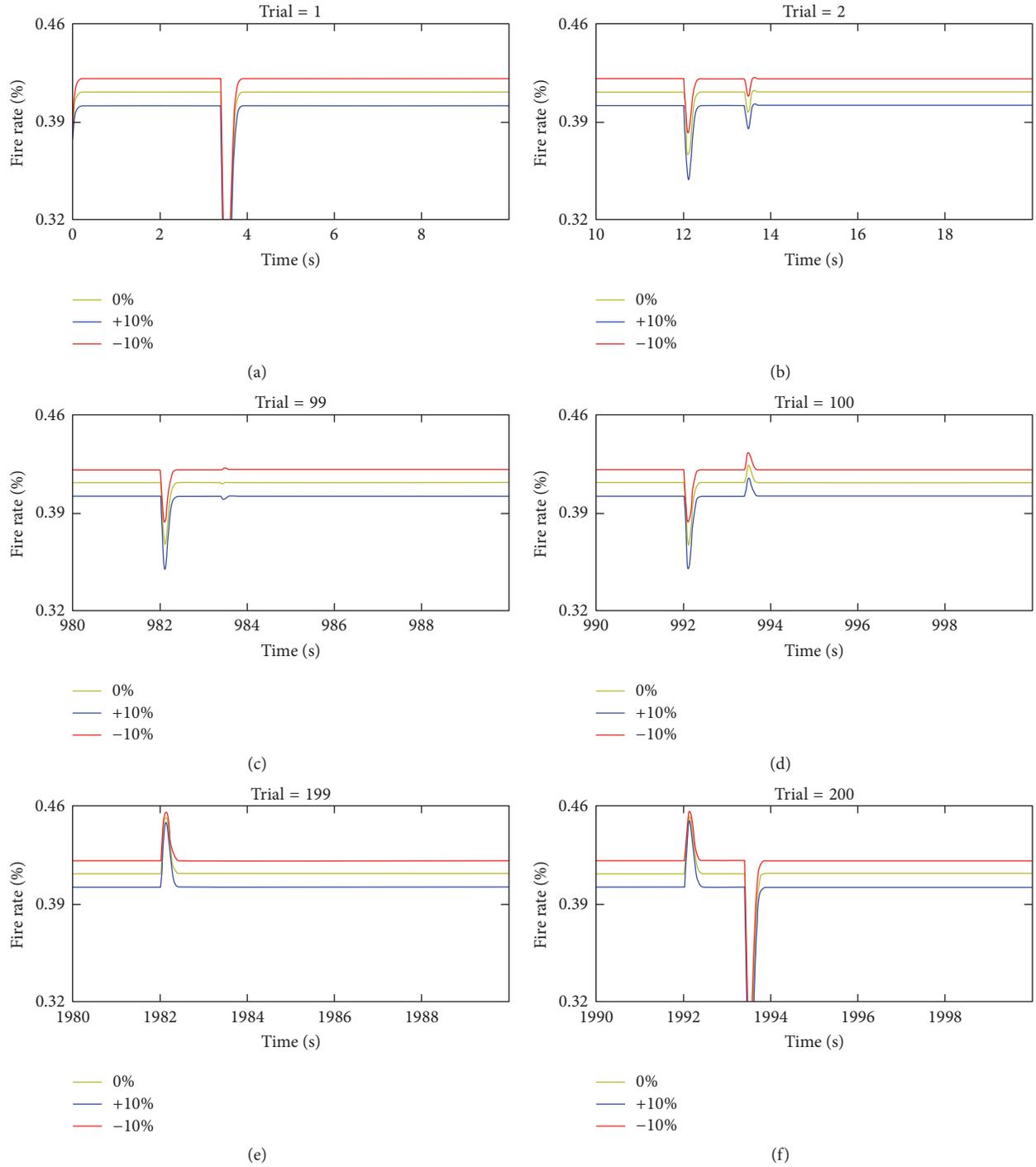


FIGURE 10: The phasic activity of LHB neurons given different weights of synapses from VP to GPb. Yellow lines indicate the activity of LHB neurons when W_{VPG} equals 1.00 and \bar{D} equals 0.19431, blue lines indicate the activity when W_{VPG} equals 1.10 and \bar{D} equals 0.20307, and red lines indicate the activity when W_{VPG} equals 0.90 and \bar{D} equals 0.18608. (a) Trial 1: phasic dip activity responds to unconditional reward. (b) Trial 2: the phasic activity shifts to the cue. (c) Trial 99: the phasic activity upon the cue and baseline activity upon the reward. (d) Trial 100: the peak activity upon reward omission. (e) Trial 199: the peak activity upon nonrewarding cue. (f) Trial 200: the dip activity upon unexpected reward.

in which DA can modulate the corticostriatal synapses and the corticostriosome synapses. This will in turn affect the DA responses, closing the loop. After learning, the weights of these synapses stabilize and remain unchanged. This led to the emergent phasic activity profiles of the nuclei in

the circuit, with the parallel pathways balancing out one another. In addition, we found striosome to be a key brain nucleus which remembers the timing of previous rewards and encodes the predicted rewards. In fact, there are recent experimental works [51] that support our model prediction.

In our model, we predict neurons in the striosome to encode expected reward, but there are alternative theories. For example, Cohen et al. [52] found that there were three types of VTA neurons and VTA GABAergic neurons may signal expected reward, which could be a key variable for dopaminergic neurons to calculate reward-prediction error. Recent works [53–55] highlight the importance of VTA GABAergic neurons. Averbeck and Costa [56] proposed that the amygdala can learn and represent expected values like the striatum, and they predicted that the amygdala may play a central role in reinforcement learning and the ventral striatum may play less of a primary role. Wagner et al. [57] suggested that the cerebellar granule cells may encode the expectation of reward. Luo et al. [58], Li et al. [59], and Hayashi et al. [60] found that serotonin neurons in the dorsal raphe nucleus can encode reward signals. Some physiological and theoretical works [17, 18, 61–63] focus on D1 and D2 receptors in the ventral striatum and suggested that they play an important role in computing reward-prediction error. Future neural circuit modeling effort would need to incorporate such findings.

To obtain the results consistent with experiments, we have adopted several assumptions. First, we assumed that the striatal neurons excite the PPTN and ventral pallidum. Striatal neurons are usually identified as GABAergic and inhibitory, but they may excite downstream neurons through disinhibitory effect or substance P released by striatal neurons [29, 30]. In fact, it has been demonstrated that substance P mediates the excitatory interaction between striatal neurons to VP neurons [29] and striatal projection neurons [30]. Second, we hypothesized that the striosome projects to the GPi which in turn projects to the GPb. Although we have no direct evidence, Hong and Hikosaka [21] have observed that typical GPe and GPi neurons are first inhibited by striatal stimulation and GPb neurons are often (but not always) excited by striatal stimulation. They proposed that inputs to GPb were mediated through inhibitory axon collaterals within the striatum [28] or GPe [24].

While developing the model, we have tried to add minimal features to the previous model of Brown et al. [16]. Hence, it is worthy of note that we have ignored several factors to simplify the model. Specifically, we ignored the connections between some brain nuclei, such as the cortex-to-GPb [2], VP-to-RMTg [3], Lhb-to-Lhb, cortex-to-Lhb [48], and DA-to-striatum [64] pathways. We also did not consider the direct Lhb-to-VTA [65] and VTA-to-Lhb [66] connections in our simulation, but we mimicked the overall inhibition of Lhb on VTA. We have also ignored the different types of activity of many brain nuclei. For instance, studies have suggested three types of GPb neurons: reward-positive type, reward-negative type, and direction selective type [2]. Our model only considers the reward-negative type since the majority of the neurons of Gpb are of the reward-negative type and this type of neurons may play a key role in reward-related information transmission.

Despite the assumptions in the model, our neural circuit model can still implement the computation for reward-based phasic signaling and reinforcement learning, as observed in a variety of experiments. The phasic activities in multiple

brain regions represent prediction error signals, which not only associates the cue with outcome but also memorizes the specific time interval between the two. This requires the neural system to hold the information predicted by the cue, compare the information with the outcome, and report the result of the comparison. In our model, the time spectrum of the striosome and the parallel excitatory and inhibitory pathways provided the platform for such computation. The peak activity of DA and Lhb neurons functions in complementary roles, encoding reward and nonreward/punishment information separately and alleviating any flooring (limiting) effect of the dip in activity of either neuron type. Our novel neural circuit model with parallel pathways provides an instantiation of such complex neural computation.

5. Mathematics and Equations

This section lists the mathematical equations of the model (Figure 2). We give the model circuit different inputs to simulate different conditions. We use differential equations to simulate the firing rates (or the activity levels) of the neurons in different brain nuclei. The model variables are summarized in Table 1, the fixed parameters are summarized in Table 2, and the mathematical expressions are below.

(i) *Different Inputs in Each Trial (Figure 2)*. The cortex, especially the orbitofrontal cortex (OFC), encodes the expectation future outcome and their response reflects the value conveyed by the combination of reward and punishment of the cue [36, 37]. Furthermore, OFC neurons fired most strongly for cues that predict large reward or small penalty and least strongly for cues that predict large penalty or small reward relative to neutral conditions [32, 33]. Therefore, we set a larger value for rewarding cue and smaller but positive value for nonrewarding cue as follows.

Reward CS input is as follows:

$$I_{C\text{-reward}} = \begin{cases} \text{background}_{IC} & 0 \leq t \leq 2 \\ \text{background}_{IC} + 0.60 & 2 < t \leq 3.60 \\ \text{background}_{IC} + 0.60e^{-(1/\tau)(t-3.60)} & t > 3.60. \end{cases} \quad (1)$$

We set $\text{background}_{IC} = 0.30$ and $\tau = 20$.

When the network receives a reward CS, the inputs from the cortex increase abruptly and last until the time when the expected reward should be given. Then, the inputs decay exponentially to baseline activity level.

Nonreward CS input is as follows:

$$I_{C\text{-nonreward}} = \begin{cases} \text{background}_{IC} & 0 \leq t \leq 2 \\ \text{background}_{IC} - 0.20 & 2 < t \leq 3.60 \\ \text{background}_{IC} - 0.20e^{-(1/\tau)(t-3.60)} & t > 3.60. \end{cases} \quad (2)$$

Reward US input is as follows:

$$I_{R\text{-reward}} = \begin{cases} \text{background}_{IR} & 0 \leq t \leq 3.40 \\ \text{background}_{IR} - 0.80 & 3.40 < t \leq 3.60 \\ \text{background}_{IR} - 0.80e^{-(1/\tau)(t-3.60)} & t > 3.60. \end{cases} \quad (3)$$

We set $\text{background}_{IR} = 0.20$.

When the network receives a reward US, the inputs from the lateral hypothalamus increase abruptly and last for a very short duration. Then, the inputs decay exponentially to baseline activity level.

Nonreward US input is as follows:

$$I_{R\text{-nonreward}} = \text{background}_{IR}. \quad (4)$$

If the network does not get reward or gets nonreward (aversion or punishment), we assume the inputs in this trial do not change, and the inputs remain at baseline level.

(ii) *Differential Equations.* First, the changes of activation level of ventral striatal cells (S) are governed by [16]

$$\frac{1}{\tau_S} \frac{d}{dt} S = -S + (1 - S) \left[\sum_i I_i W_{iS} + I_R W_{RS} \right]. \quad (5)$$

The activity level of striatal cells changes in the wake of its passive decay and excitation from CS inputs and US inputs. The weight W_{RS} is fixed while the weight W_{iS} can be changed.

The weight W_{iS} is governed by [17, 18]

$$\begin{aligned} \frac{1}{\tau_{WS}} \frac{d}{dt} W_{iS} \\ = G_{WS} S [\alpha_{WS} N^+ I_i (C_{WS}^{\max} - W_{iS}) - \beta_{WS} N^- W_{iS}]. \end{aligned} \quad (6)$$

The synaptic weight changes are induced by phasic dopamine burst or dip signal, N^+ and N^- (defined in (7) and (8)). Learning is gated by delayed release of a second messenger and calcium signal G_{WS} is governed by (9) and (11) at a rate $r = 12.5$.

The positive reinforcement-learning signal N^+ derives from excitatory phasic fluctuations of the dopamine signal above the baseline:

$$N^+ = [D - \bar{D} - \Gamma_D]^+. \quad (7)$$

The complementary negative reinforcement-learning signal N^- derives from inhibitory phasic fluctuations of the dopamine signal below baseline:

$$N^- = [\bar{D} - D - \Gamma_N]^+. \quad (8)$$

Second, striosomes play an important role in the phasic activities of DA neurons and LHB neurons because of its timing spectrum mechanism: a spectrum of striosomal MSPN

second messenger activities x_{ij} responds to the i th input at rates r_j :

$$\frac{d}{dt} x_{ij} = r_j [-x_{ij} + (1 - x_{ij}) I_i], \quad (9)$$

where the second messenger buildup rates are given by

$$r_j = \frac{\alpha_r}{\beta_r + j}. \quad (10)$$

The activities x_{ij} induce intracellular calcium dynamics within a given spine (j) at delays determined by r_j . The intracellular calcium spike is represented by the quantity $[G_{ij} Y_{ij}]^+$, where

$$\frac{d}{dt} G_{ij} = \alpha_G (B_G - G_{ij}) f_G(x_{ij} - \Gamma_G) - \beta_G G_{ij}, \quad (11)$$

$$\frac{d}{dt} Y_{ij} = \alpha_Y (1 - Y_{ij}) - \beta_Y [G_{ij} Y_{ij} - \Gamma_Y]^+. \quad (12)$$

In (11), $f_G(x)$ is a step function:

$$f_G(x) = \begin{cases} 1 & (x > 0) \\ 0 & (x < 0). \end{cases} \quad (13)$$

In the brief interval when the calcium concentration at a particular spine exceeds a threshold activity Γ_S , CS-striosomal weight Z_{ij} at that particular spine becomes eligible for change that may be induced by dopaminergic bursts (N^+) or dips (N^-).

$$\begin{aligned} \frac{d}{dt} Z_{ij} \\ = \alpha_Z [G_{ij} Y_{ij} - \Gamma_S]^+ ((A_Z - Z_{ij}) N^+ - B_Z Z_{ij} N^-). \end{aligned} \quad (14)$$

Third, the changes in the level of PPTN (P) are described by the following differential equations:

$$\frac{1}{\tau_{P1}} \frac{d}{dt} P_{\text{pre-excite}} = -P_{\text{pre-excite}} + (1 - P_{\text{pre-excite}}) W_{SP} S \quad (15)$$

$$\frac{1}{\tau_{P2}} \frac{d}{dt} P_{\text{pre-inhibit}} = -P_{\text{pre-inhibit}} + (1 - P_{\text{pre-inhibit}}) W_{SP} S \quad (16)$$

$$\frac{1}{\tau_P} \frac{d}{dt} P = \text{background}_P - P \quad (17)$$

$$+ (1 - P) W_P \text{input}_{\text{pre},P},$$

where

$$\begin{aligned} \text{input}_{\text{pre},P} \\ = \begin{cases} [P_{\text{pre-excite}} - P_{\text{pre-inhibit}} - \Gamma_{P12}]^+ & P_{\text{pre-excite}} > P_{\text{pre-inhibit}} \\ -[P_{\text{pre-inhibit}} - P_{\text{pre-excite}} - \Gamma_{P12}]^+ & P_{\text{pre-excite}} < P_{\text{pre-inhibit}} \end{cases} \end{aligned} \quad (18)$$

$P_{\text{pre-excite}}$ and $P_{\text{pre-inhibit}}$ can be regarded as the effect of substance P and GABA on PPTN. Ventral striatum neurons can secrete substance P and GABA. Substance P excites

the following neurons, while GABA inhibits the following neurons; $\text{input}_{\text{pre},P}$ denotes the net effect of substance P and GABA. The authors believe that this explanation is more realistic, but it needs more physiological experiments to be testified. The changes of the activity level of PPTN neurons depend on the background inputs, its decay, and the net effect from the striatum.

Fourth, the changes in the level of ventral pallidum (VP) are described by the following differential equations:

$$\begin{aligned} \frac{1}{\tau_{VP1}} \frac{d}{dt} VP_{\text{pre-excite}} &= -VP_{\text{pre-excite}} \\ &+ (1 - VP_{\text{pre-excite}}) W_{SVP} S \end{aligned}$$

$$\begin{aligned} \frac{1}{\tau_{VP2}} \frac{d}{dt} VP_{\text{pre-inhibit}} &= -VP_{\text{pre-inhibit}} \\ &+ (1 - VP_{\text{pre-inhibit}}) W_{SVP} S \\ \frac{1}{\tau_{VP}} \frac{d}{dt} VP &= \text{background}_{VP} - VP \\ &+ (1 - VP) W_{VP} \text{input}_{\text{pre},VP}, \end{aligned} \quad (19)$$

where

$$\text{input}_{\text{pre},VP} = \begin{cases} [VP_{\text{pre-excite}} - VP_{\text{pre-inhibit}} - \Gamma_{VP12}]^+ & VP_{\text{pre-excite}} > VP_{\text{pre-inhibit}} \\ -[VP_{\text{pre-inhibit}} - VP_{\text{pre-excite}} - \Gamma_{VP12}]^+ & VP_{\text{pre-excite}} < VP_{\text{pre-inhibit}} \end{cases} \quad (20)$$

The explanation is similar to (15)~(18). The changes of the activity level of VP neurons result from the background inputs, its decay, and the net effect from the striatum.

Fifth, changes in the level of GPb neurons are described by the following differential equation:

$$\begin{aligned} \frac{1}{\tau_{GPb}} \frac{d}{dt} GPb &= \text{background}_{GPb} - GPb + (1 - GPb) \\ &\cdot \left(W_{SOG} \sum_{i,j} [G_{ij} Y_{ij} - \Gamma_S]^+ Z_{ij} - W_{VPG} VP \right). \end{aligned} \quad (21)$$

The changes of the activity level of GPb neurons are determined by the background inputs, its decay, and the inhibitory effect from VP neurons and the disinhibitory input from striosomes.

Sixth, changes in the level of LHb neural activity are described by the following differential equation:

$$\begin{aligned} \frac{1}{\tau_{LHb}} \frac{d}{dt} LHb &= \text{background}_{LHb} - LHb \\ &+ (1 - LHb) W_{GL} [GPb - \Gamma_{GPb}]^+. \end{aligned} \quad (22)$$

The changes of the activity level of LHb neurons result from the background inputs, its decay, and the excitatory input from the GPb.

Seventh, changes in the level of RMTg neurons are described by the following differential equation:

$$\begin{aligned} \frac{1}{\tau_{RMTg}} \frac{d}{dt} RMTg &= \text{background}_{RMTg} - RMTg \\ &+ (1 - RMTg) W_{LR} [LHb - \Gamma_{LHb}]^+. \end{aligned} \quad (23)$$

The changes of the activity level of RMTg neurons depend on the background inputs, its decay, and the excitatory input from the LHb.

Eighth, changes in the level of dopaminergic neurons (D) are described by the following differential equation:

$$\begin{aligned} \frac{1}{\tau_D} \frac{d}{dt} D &= \text{background}_D - D \\ &+ (1 - D) (W_{PD} [P - \Gamma_P]^+ - W_{RD} RMTg) \\ &- (D + h_D) \sum_{i,j} [G_{ij} Y_{ij} - \Gamma_S]^+ Z_{ij}. \end{aligned} \quad (24)$$

The changes of the activity level of dopaminergic neurons depend on the background inputs, its decay, the inhibitory effect from RMTg neurons and striosomes, and the excitatory input from the PPTN.

Conflicts of Interest

The authors declare that they have no conflicts of interest regarding this work.

Acknowledgments

Da-Hui Wang was supported by NSFC under Grants nos. 31271169 and 31671077, the Fundamental Research Funds for the Central Universities, and BMSTC (Beijing Municipal Science and Technology Commission) under Grant no. Z171100000117007. KongFatt Wong-Lin was supported by BBSRC (BB/P003427/1), COST Action Open Multiscale Systems Medicine (OpenMultiMed) supported by COST (European Cooperation in Science and Technology), and Northern Ireland Functional Brain Mapping Facility (1303/101154803) funded by Invest NI and the University of Ulster. Da-Hui Wang and KongFatt Wong-Lin were also supported by the Royal Society-NSFC International Exchanges Scheme-Cost Share Programme (31511130066, IE141307).

References

- [1] C. D. Fiorillo, P. N. Tobler, and W. Schultz, "Discrete coding of reward probability and uncertainty by dopamine neurons," *Science*, vol. 299, no. 5614, pp. 1898–1902, 2003.
- [2] S. Hong and O. Hikosaka, "The Globus Pallidus Sends Reward-Related Signals to the Lateral Habenula," *Neuron*, vol. 60, no. 4, pp. 720–729, 2008.
- [3] S. Hong, T. C. Jhou, M. Smith, K. S. Saleem, and O. Hikosaka, "Negative reward signals from the lateral habenula to dopamine neurons are mediated by rostromedial tegmental nucleus in primates," *The Journal of Neuroscience*, vol. 31, no. 32, pp. 11457–11471, 2011.
- [4] C. M. Kuhnen and B. Knutson, "The neural basis of financial risk taking," *Neuron*, vol. 47, no. 5, pp. 763–770, 2005.
- [5] M. Matsumoto and O. Hikosaka, "Lateral habenula as a source of negative reward signals in dopamine neurons," *Nature*, vol. 447, no. 7148, pp. 1111–1115, 2007.
- [6] A. N. McCoy and M. L. Platt, "Risk-sensitive neurons in macaque posterior cingulate cortex," *Nature Neuroscience*, vol. 8, no. 9, pp. 1220–1227, 2005.
- [7] I. E. Monosov and O. Hikosaka, "Selective and graded coding of reward uncertainty by neurons in the primate anterodorsal septal region," *Nature Neuroscience*, vol. 16, no. 6, pp. 756–762, 2013.
- [8] W. Schultz, P. Dayan, and P. R. Montague, "A neural substrate of prediction and reward," *Science*, vol. 275, no. 5306, pp. 1593–1599, 1997.
- [9] E. Van Duuren, G. Van Der Plasse, J. Lankelma, R. N. J. M. A. Joosten, M. G. P. Feenstra, and C. M. A. Pennartz, "Single-cell and population coding of expected reward probability in the orbitofrontal cortex of the rat," *The Journal of Neuroscience*, vol. 29, no. 28, pp. 8965–8976, 2009.
- [10] E. S. Bromberg-Martin, M. Matsumoto, and O. Hikosaka, "Dopamine in motivational control: rewarding, aversive, and alerting," *Neuron*, vol. 68, no. 5, pp. 815–834, 2010.
- [11] M. Matsumoto and O. Hikosaka, "Two types of dopamine neuron distinctly convey positive and negative motivational signals," *Nature*, vol. 459, no. 7248, pp. 837–841, 2009.
- [12] "Correction for Glimcher, Understanding dopamine and reinforcement learning: The dopamine reward prediction error hypothesis," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, no. 42, pp. 17568–17569, 2011.
- [13] Y. Niv, M. O. Duff, and P. D. Dayan, "Dopamine, uncertainty and TD learning," *Behavioral and Brain Functions*, vol. 1, 2005.
- [14] R. S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1988.
- [15] R. S. Sutton and A. G. Barto, "Toward a modern theory of adaptive networks: Expectation and prediction," *Psychological Review*, vol. 88, no. 2, pp. 135–170, 1981.
- [16] J. Brown, D. Bullock, and S. Grossberg, "How the basal ganglia use parallel excitatory and inhibitory learning pathways to selectively respond to unexpected rewarding cues," *The Journal of Neuroscience*, vol. 19, no. 23, pp. 10502–10511, 1999.
- [17] O. T. Can and D. Bullock, "A local circuit model of learned striatal and dopamine cell responses under probabilistic schedules of reward," *The Journal of Neuroscience*, vol. 28, no. 40, pp. 10062–10074, 2008.
- [18] C. O. Tan and D. Bullock, "A dopamine-acetylcholine cascade: Simulating learned and lesion-induced behavior of striatal cholinergic interneurons," *Journal of Neurophysiology*, vol. 100, no. 4, pp. 2409–2421, 2008.
- [19] O. Hikosaka, "The habenula: from stress evasion to value-based decision-making," *Nature Reviews Neuroscience*, vol. 11, no. 7, pp. 503–513, 2010.
- [20] T. C. Jhou, H. L. Fields, M. G. Baxter, C. B. Saper, and P. C. Holland, "The Rostromedial Tegmental Nucleus (RMTg), a GABAergic Afferent to Midbrain Dopamine Neurons, Encodes Aversive Stimuli and Inhibits Motor Responses," *Neuron*, vol. 61, no. 5, pp. 786–800, 2009.
- [21] S. Hong and O. Hikosaka, "Diverse sources of reward value signals in the basal ganglia nuclei transmitted to the lateral habenula in the monkey," *Frontiers in Human Neuroscience*, Article ID 778, 2013.
- [22] P. Kalén, M. Pritzel, A. Nieoullon, and L. Wiklund, "Further evidence for excitatory amino acid transmission in the lateral habenular projection to the rostral raphe nuclei: Lesion-induced decrease of high affinity glutamate uptake," *Neuroscience Letters*, vol. 68, no. 1, pp. 35–40, 1986.
- [23] K. Brinshawitz, A. Dittgen, V. I. Madai, R. Lommel, S. Geisler, and R. W. Veh, "Glutamatergic axons from the lateral habenula mainly terminate on GABAergic neurons of the ventral mid-brain," *Neuroscience*, vol. 168, no. 2, pp. 463–476, 2010.
- [24] A. R. Sadek, P. J. Magill, and J. P. Bolam, "A single-cell analysis of intrinsic connectivity in the rat globus pallidus," *The Journal of Neuroscience*, vol. 27, no. 24, pp. 6352–6362, 2007.
- [25] S. J. Shabel, C. D. Proulx, A. Trias, R. T. Murphy, and R. Malinow, "Input to the Lateral Habenula from the Basal Ganglia Is Excitatory, Aversive, and Suppressed by Serotonin," *Neuron*, vol. 74, no. 3, pp. 475–481, 2012.
- [26] S. J. Shabel, C. D. Proulx, J. Piriz, and R. Malinow, "GABA/glutamate co-release controls habenula output and is modified by antidepressant treatment," *Science*, vol. 345, no. 6203, pp. 1494–1498, 2014.
- [27] N. Rajakumar, K. Elisevich, and B. A. Flumerfelt, "Compartamental origin of the striato-entopeduncular projection in the rat," *Journal of Comparative Neurology*, vol. 331, no. 2, pp. 286–296, 1993.
- [28] L. Tremblay and M. Filion, "Responses of pallidal neurons to striatal stimulation in intact waking monkeys," *Brain Research*, vol. 498, no. 1, pp. 1–16, 1989.
- [29] T. C. Napier, I. Mitrovic, L. Churchill, M. A. Klitenick, X.-Y. Lu, and P. W. Kalivas, "Substance P in the ventral pallidum: Projection from the ventral striatum, and electrophysiological and behavioral consequences of pallidal substance P," *Neuroscience*, vol. 69, no. 1, pp. 59–70, 1995.
- [30] C. P. Blomeley, L. A. Kehoe, and E. Bracci, "Substance p mediates excitatory interactions between striatal projection neurons," *The Journal of Neuroscience*, vol. 29, no. 15, pp. 4953–4963, 2009.
- [31] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes*, Cambridge University Press, Cambridge, UK, 2nd edition, 1992.
- [32] S. E. Morrison and C. D. Salzman, "Representations of appetitive and aversive information in the primate orbitofrontal cortex," *Annals of the New York Academy of Sciences*, vol. 1239, no. 1, pp. 59–70, 2011.
- [33] M. R. Roesch and C. R. Olson, "Neuronal Activity Related to Reward Value and Motivation in Primate Frontal Cortex," *Science*, vol. 304, no. 5668, pp. 307–310, 2004.
- [34] S. Fusi, E. K. Miller, and M. Rigotti, "Why neurons mix: High dimensionality for higher cognition," *Current Opinion in Neurobiology*, vol. 37, pp. 66–74, 2016.

- [35] V. Mante, D. Sussillo, K. V. Shenoy, and W. T. Newsome, "Context-dependent computation by recurrent dynamics in prefrontal cortex," *Nature*, vol. 503, no. 7474, pp. 78–84, 2013.
- [36] C. Padoa-Schioppa and J. A. Assad, "Neurons in the orbitofrontal cortex encode economic value," *Nature*, vol. 441, no. 7090, pp. 223–226, 2006.
- [37] C. Padoa-Schioppa and K. E. Conen, "Orbitofrontal Cortex: A Neural Circuit for Economic Decisions," *Neuron*, vol. 96, no. 4, pp. 736–754, 2017.
- [38] V. Pascoli, J. Terrier, A. Hiver, and C. Lüscher, "Sufficiency of Mesolimbic Dopamine Neuron Stimulation for the Progression to Addiction," *Neuron*, vol. 88, no. 5, pp. 1054–1066, 2015.
- [39] P. E. M. Phillips, D. L. Robinson, G. D. Stuber, R. M. Carelli, and R. M. Wightman, "Real-time measurements of phasic changes in extracellular dopamine concentration in freely moving rats by fast-scan cyclic voltammetry," *Methods in Molecular Medicine*, vol. 79, pp. 443–464, 2003.
- [40] E. K. Miller, C. A. Erickson, and R. Desimone, "Neural mechanisms of visual working memory in prefrontal cortex of the macaque," *The Journal of Neuroscience*, vol. 16, no. 16, pp. 5154–5167, 1996.
- [41] T. Ljungberg, P. Apicella, and W. Schultz, "Responses of monkey dopamine neurons during learning of behavioral reactions," *Journal of Neurophysiology*, vol. 67, no. 1, pp. 145–163, 1992.
- [42] W. Schultz, "Predictive reward signal of dopamine neurons," *Journal of Neurophysiology*, vol. 80, no. 1, pp. 1–27, 1998.
- [43] W.-X. Pan, R. Schmidt, J. R. Wickens, and B. I. Hyland, "Dopamine cells respond to predicted events during classical conditioning: Evidence for eligibility traces in the reward-learning network," *The Journal of Neuroscience*, vol. 25, no. 26, pp. 6235–6242, 2005.
- [44] C. J. Burke and P. N. Tobler, "Time, Not Size, Matters for Striatal Reward Predictions to Dopamine," *Neuron*, vol. 91, no. 1, pp. 8–11, 2016.
- [45] J. C. Fiala, S. Grossberg, and D. Bullock, "Metabotropic glutamate receptor activation in cerebellar purkinje cells as substrate for adaptive timing of the classically conditioned eye-blink response," *The Journal of Neuroscience*, vol. 16, no. 11, pp. 3760–3774, 1996.
- [46] H. M. Bayer and P. W. Glimcher, "Midbrain dopamine neurons encode a quantitative reward prediction error signal," *Neuron*, vol. 47, no. 1, pp. 129–141, 2005.
- [47] G. Morris, D. Arkadir, A. Nevet, E. Vaadia, and H. Bergman, "Coincident but distinct messages of midbrain dopamine and striatal tonically active neurons," *Neuron*, vol. 43, no. 1, pp. 133–143, 2004.
- [48] F. J. Meye, S. Lecca, K. Valentinova, and M. Mameli, "Synaptic and cellular profile of neurons in the lateral habenula," *Frontiers in Human Neuroscience*, vol. 7, 2013.
- [49] M. Song, Y. S. Jo, Y.-K. Lee, and J.-S. Choi, "Lesions of the lateral habenula facilitate active avoidance learning and threat extinction," *Behavioural Brain Research*, vol. 318, pp. 12–17, 2017.
- [50] C. L. Danna, P. D. Shepard, and G. I. Elmer, "The habenula governs the attribution of incentive salience to reward predictive cues," *Frontiers in Human Neuroscience*, vol. 7, 2013.
- [51] Y. K. Takahashi, A. J. Langdon, Y. Niv, and G. Schoenbaum, "Temporal Specificity of Reward Prediction Errors Signaled by Putative Dopamine Neurons in Rat VTA Depends on Ventral Striatum," *Neuron*, vol. 91, no. 1, pp. 182–193, 2016.
- [52] J. Y. Cohen, S. Haesler, L. Vong, B. B. Lowell, and N. Uchida, "Neuron-type-specific signals for reward and punishment in the ventral tegmental area," *Nature*, vol. 482, no. 7383, pp. 85–88, 2012.
- [53] M. Morales and E. B. Margolis, "Ventral tegmental area: Cellular heterogeneity, connectivity and behaviour," *Nature Reviews Neuroscience*, vol. 18, no. 2, pp. 73–85, 2017.
- [54] W. R. Stauffer, "Systems Neuroscience: Shaping the Reward Prediction Error Signal," *Current Biology*, vol. 25, no. 22, pp. R1081–R1084, 2015.
- [55] J. H. Yoo, V. Zell, N. Gutierrez-Reed et al., "Ventral tegmental area glutamate neurons co-release GABA and promote positive reinforcement," *Nature Communications*, vol. 7, Article ID 13697, 2016.
- [56] B. B. Averbeck and V. D. Costa, "Motivational neural circuits underlying reinforcement learning," *Nature Neuroscience*, vol. 20, no. 4, pp. 505–512, 2017.
- [57] M. J. Wagner, T. H. Kim, J. Savall, M. J. Schnitzer, and L. Luo, "Cerebellar granule cells encode the expectation of reward," *Nature*, vol. 544, no. 7648, pp. 96–100, 2017.
- [58] M. Luo, J. Zhou, and Z. Liu, "Reward processing by the dorsal raphe nucleus: 5-HT and beyond," *Learning & Memory*, vol. 22, no. 9, pp. 452–460, 2015.
- [59] Y. Li, W. Zhong, D. Wang et al., "Serotonin neurons in the dorsal raphe nucleus encode reward signals," *Nature Communications*, vol. 7, Article ID 10503, 2016.
- [60] K. Hayashi, K. Nakao, and K. Nakamura, "Appetitive and aversive information coding in the primate dorsal raphe nucleus," *The Journal of Neuroscience*, vol. 35, no. 15, pp. 6195–6208, 2015.
- [61] T. Hikida, M. Morita, and T. Macpherson, "Neural mechanisms of the nucleus accumbens circuit in reward and aversive learning," *Neuroscience Research*, vol. 108, pp. 1–5, 2016.
- [62] M. D. Humphries and T. J. Prescott, "The ventral basal ganglia, a selection mechanism at the crossroads of space, strategy, and reward," *Progress in Neurobiology*, vol. 90, no. 4, pp. 385–417, 2010.
- [63] R. Keiflin and P. H. Janak, "Dopamine Prediction Errors in Reward Learning and Addiction: From Theory to Neural Circuitry," *Neuron*, vol. 88, no. 2, pp. 247–263, 2015.
- [64] N. F. Parker, C. M. Cameron, J. P. Taliaferro et al., "Reward and choice encoding in terminals of midbrain dopamine neurons depends on striatal target," *Nature Neuroscience*, 2016.
- [65] W. C. Poller, V. I. Madai, R. Bernard, G. Laube, and R. W. Veh, "A glutamatergic projection from the lateral hypothalamus targets VTA-projecting neurons in the lateral habenula of the rat," *Brain Research*, vol. 1507, pp. 45–60, 2013.
- [66] A. M. Stamatakis, J. H. Jennings, R. L. Ung et al., "A Unique Population of Ventral Tegmental Area Neurons Inhibits the Lateral Habenula to Promote Reward," *Neuron*, vol. 80, no. 4, pp. 1039–1053, 2013.

Research Article

Adaptive Neural Networks Control Using Barrier Lyapunov Functions for DC Motor System with Time-Varying State Constraints

Lei Ma¹ and Dapeng Li ²

¹College of Science, Liaoning University of Technology, Jinzhou, Liaoning 121001, China

²School of Electrical Engineering, Liaoning University of Technology, Jinzhou 121001, China

Correspondence should be addressed to Dapeng Li; li_dapengsir@163.com

Received 28 June 2017; Accepted 16 November 2017; Published 31 January 2018

Academic Editor: Yanan Li

Copyright © 2018 Lei Ma and Dapeng Li. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper proposes an adaptive neural network (NN) control approach for a direct-current (DC) system with full state constraints. To guarantee that state constraints always remain in the asymmetric time-varying constraint regions, the asymmetric time-varying Barrier Lyapunov Function (BLF) is employed to structure an adaptive NN controller. As we all know that the constant constraint is only a special case of the time-varying constraint, hence, the proposed control method is more general for dealing with constraint problem as compared with the existing works on DC systems. As far as we know, this system is the first studied situations with time-varying constraints. Using Lyapunov analysis, all signals in the closed-loop system are proved to be bounded and the constraints are not violated. In this paper, the effectiveness of the control method is demonstrated by simulation results.

1. Introduction

Due to the requirements of practice and the development of theory, the controller design of uncertain system has become a new research direction and attracted more and more scholars' attention. The uncertainty of the actual engineering system has been studied in many works [1–4]. The neural networks [5] and fuzzy logic systems [6] have become the two main tools which can effectively deal with the unknown functions in the systems. In [7, 8], these are studies of some actual engineering systems with uncertain parameters. In [9, 10], the NN is used to approximate several random perturbations and unknown functions. In [11–16], several nonlinear system solutions are studied based on neural networks and fuzzy logic systems. In [17], adaptive control schemes based on neural networks were proposed for nonlinear systems with unknown functions. Based on neural networks and fuzzy logic systems, the significant studies proposed the novel adaptive tracking control methods for nonlinear SISO systems in [18–20] and MIMO systems in [21–23]. However,

it is worth noting that the constraint problem is worth noting in the above approaches, which lead to the inaccuracy or oscillations of the engineering systems and even cause control systems instability.

In fact, there are constraints in most physical systems with various forms, for example, physical stoppages, saturation, performance, and safety specifications, such as restricted robot manipulation system [24], application to chemical process [25], networked surveillance robots systems [26], and nonuniform gantry crane [27]. In recent years, the barrier Lyapunov functions become the main tools to solve the constrained problem which was proposed for the first time in [28]. Based on BLF, some adaptive control methods were presented for nonlinear systems with output constant constraint in [29, 30] and state constant constraint in [31–34]. As we known, the constant constraint is the special case of the time-varying constraint. Subsequently, the authors in [35, 36] proposed some adaptive control approaches to address the stability problem of nonlinear systems with time-varying constraints.

Motor is the most important electromechanical energy conversion device, which has been widely used in the industrial and agricultural production, transportation, aerospace, and so on. In particular, the motor system with unknown uncertainties has attracted the attention of many scholars, and the control problem of the motor system becomes more and more important. In [37, 38], the authors proposed two adaptive control methods for systems with unknown functions. The authors in [39, 40] presented an adaptive control with time-varying output constraints for DC motor systems. According to the above descriptions, the urgent problem is how to address the stability problem of the DC motor system with time-varying state constraints.

This paper presents an adaptive NN tracking control method for DC motor systems with time-varying state constraints. As far as we know, there is no work dealing with such DC motor systems in the literature at present stage. The contributions of this paper are summarized as follows. (1) The time-varying state constraints are first considered in the DC motor systems; comparing with the existing on DC motor systems, the proposed control method is more general and extensive in the engineering field. (2) To guarantee that the state constraints always remain in the time-varying constrained sets, the asymmetric time-varying BLF is utilized. (3) A novel adaptive tracking controller based on the neural networks and backstepping technique is structured to guarantee that all signals in the closed-loop system are bounded, the tracking errors converge to a small neighborhood of zero and the time-varying state constraints are not transitioned.

2. Problem Formulation and Preliminaries

Consider the dynamic system with the DC motor without vibration mode as the following form:

$$\begin{aligned} \dot{\alpha}_1 &= \alpha_2 \\ J\dot{\alpha}_2 + f\dot{\alpha}_1 + T_f + d &= u \\ y &= \alpha_1, \end{aligned} \quad (1)$$

where $\alpha_1(t)$ is the motor angular position; α_2 stands for motor angular velocity; J is a known inertia, f is an unmeasured viscous friction, and T_f is an unmeasured nonlinear friction; $d(t)$ represents the unknown disturbance but bounded with $\|d(t)\| \leq d_M$; $y \in R$ is the system output; and u represents the motor torque. In particular, output $y(t)$ is required as follows:

$$\underline{k}_{c_1}(t) < y(t) < \bar{k}_{c_1}(t), \quad \forall t \geq 0, \quad (2)$$

where $\bar{k}_{c_1} : R_+ \rightarrow R$ and $\underline{k}_{c_1} : R_+ \rightarrow R$ such that $\bar{k}_{c_1}(t) > \underline{k}_{c_1}(t)$, $\forall t \in R_+$.

Remark 1. From (2), the states of DC systems are constrained by the considered time-varying functions. In [35, 36], the constraint problem is omitted, which is the main factor of the oscillations of the engineering systems. The authors

in [39] addressed the stability problem of DC motor systems with constant constraint which is the special case of the time-varying constraint. Comparing with the [40], the authors only consider time-varying output constraint; the proposed adaptive control method tries to stabilize the DC motor systems with time-varying state constraints, which cause the difficulty of controller design.

In this paper, the control objective is to design an adaptive NN tracking controller u which adjusts the output of DC motor systems y to track desired trajectory of the reference signal $y_d(t)$ in the range of time-varying constraint functions. Meanwhile, all signals in the closed-loop systems are bounded and the time-varying state constraints are not violated.

Assumption 2 (see [35]). There exist constants \bar{K}_{c_i} and \underline{K}_{c_i} , $i = 0, 1, \dots, n$, such that $\bar{k}_{c_1}(t) \leq \bar{K}_{c_0}$, $\underline{k}_{c_1}(t) \geq \underline{K}_{c_0}$, and $|\bar{k}_{c_1}^{(i)}(t)| \leq \bar{K}_{c_i}$, $|\underline{k}_{c_1}^{(i)}(t)| \leq \underline{K}_{c_i}$, $\forall t \geq 0$, $i = 1, \dots, n$.

Assumption 3 (see [32]). There exist functions $\bar{Y}_0 : R_+ \rightarrow R_+$ and $\underline{Y}_0 : R_+ \rightarrow R_+$ satisfying $\bar{Y}_0(t) < \bar{k}_{c_1}(t)$ and $\underline{Y}_0(t) > \underline{k}_{c_1}(t)$ $\forall t \geq 0$, and positive constants Y_i , $i = 1, \dots, n$, such that the desired trajectory $y_d(t)$ and its time derivatives satisfy $\underline{Y}_0(t) \leq y_d(t) \leq \bar{Y}_0(t)$ and $|y_d^{(i)}(t)| \leq Y_i$, $i = 1, \dots, n$, $\forall t \geq 0$.

The following lemma is represented for the establishment of binding compensation and performance limits.

Lemma 4 (see [28]). Let $Z := \{\xi \in R : |\xi| < 1\} \subset R$ and $N := R^l \times Z \subset R^{l+1}$ be open sets. Take into account the system

$$\dot{\alpha} = g(t, \alpha), \quad (3)$$

where $\alpha := [\omega, \xi]^T \in N$ and $g : R_+ \times N \rightarrow R^{l+1}$ in t is piecewise continuous and in α is locally Lipschitz, united in t , on $R_+ \times N$.

Suppose that there are functions $H : R^l \times R_+ \rightarrow R_+$ and $V_1 : Z \rightarrow R_+$. In their respective domains, they are continuously differentiable and positive definite, such that

$$\begin{aligned} V_1(\xi) &\rightarrow \infty, \quad |\xi| \rightarrow 1 \\ \lambda_1(\|\omega\|) &\leq H(\omega, t) \leq \lambda_2(\|\omega\|), \end{aligned} \quad (4)$$

where λ_1 and λ_2 are class K_∞ functions. Let $V(\alpha) := V_1(\xi) + U(\omega, t)$, and $\varepsilon(0) \in Z$. If the inequality is established:

$$\dot{V} = \frac{\partial V}{\partial \alpha} g \leq 0 \quad (5)$$

in $\xi \in Z$, $\xi(t) \in Z \forall t \in [0, \infty)$.

Lemma 5 (see [35]). For all $|\xi| < 1$ and positive integer p , the inequality $\log 1/(1 - \xi^{2p}) < \xi^{2p}/(1 - \xi^{2p})$.

Proof. For $|\xi| < 1$, the term $\xi^{2p}/(1 - \xi^{2p})$ can be rewritten as

$$\begin{aligned} \frac{\xi^{2p}}{1 - \xi^{2p}} &= \log\left(e^{\xi^{2p}/(1 - \xi^{2p})}\right) \\ &\geq \log\left[1 + \frac{\xi^{2p}}{1 - \xi^{2p}} + \sum_{n=2}^{\infty} \frac{(\xi^{2p}/(1 - \xi^{2p}))^n}{n!}\right] \\ &\geq \log\left(1 + \frac{\xi^{2p}}{1 - \xi^{2p}}\right) = \log\frac{1}{1 - \xi^{2p}}. \end{aligned} \quad (6)$$

The proof is completed. \square

3. State Feedback Adaptive Controller Designs

This paper presents an adaptive tracking controller based on a backstepping technique with the asymmetric time-varying BLF for the DC motor systems. The detailed designs process is shown in this section.

Denote $z_1 = \alpha_1 - y_d$, $z_2 = \alpha_2 - \sigma_1$, where σ_1 is the virtual controller which will be given later on. We consider the time-varying asymmetric BLF:

$$\begin{aligned} V_1 &= \frac{q(z_1)}{2p} \log \frac{k_{b_1}^{2p}(t)}{k_{b_1}^{2p}(t) - z_1^{2p}} \\ &\quad + \frac{1 - q(z_1)}{2p} \log \frac{k_{a_1}^{2p}(t)}{k_{a_1}^{2p}(t) - z_1^{2p}}, \end{aligned} \quad (7)$$

where p is a positive integer.

The time-varying barriers are chosen as

$$k_{a_1}(t) = y_d(t) - \underline{k}_{c_1}(t) \quad (8)$$

$$k_{b_1}(t) = \bar{k}_{c_1}(t) - y_d(t) \quad (9)$$

and $q(\cdot)$ is defined as

$$q(\cdot) = \begin{cases} 1, & \text{if } \cdot > 0 \\ 0, & \text{if } \cdot \leq 0. \end{cases} \quad (10)$$

Based on Assumptions 2 and 3, there are positive constants \underline{k}_{b_1} , \bar{k}_{b_1} , \underline{k}_{a_1} , and \bar{k}_{a_1} , such that

$$\begin{aligned} \underline{k}_{b_1} &\leq k_{b_1}(t) \leq \bar{k}_{b_1}, \\ \underline{k}_{a_1} &\leq k_{a_1}(t) \leq \bar{k}_{a_1}, \quad \forall t \geq 0 \end{aligned} \quad (11)$$

Through the change of error coordinates,

$$\begin{aligned} \xi_{a_i} &= \frac{z_i}{k_{a_i}}, \\ \xi_{b_i} &= \frac{z_i}{k_{b_i}}, \\ \xi_i &= q(z_i) \xi_{b_i} + (1 - q(z_i)) \xi_{a_i}, \quad i = 1, 2. \end{aligned} \quad (12)$$

According to (12), (7) can be rewritten as

$$V_1 = \frac{1}{2p} \log \frac{1}{1 - \xi_1^{2p}}. \quad (13)$$

Remark 6. According to (10), we know that when $z_1 > 0$, we obtain $q(z_1) = 1$, $\xi_1 = \xi_{b_1}$, and $V_1 = (1/2p) \log(1/(1 - \xi_{b_1}^{2p})) = (1/2p) \log(1/(1 - \xi_1^{2p}))$. When $z_1 < 0$, we obtain $q(z_1) = 0$, $\xi_1 = \xi_{a_1}$, and $V_1 = (1/2p) \log(1/(1 - \xi_{a_1}^{2p})) = (1/2p) \log(1/(1 - \xi_1^{2p}))$. From the above, we can get (13) based on (12).

Obviously, under the premise of $|\xi| < 1$, V_1 is definite continuously differentiable. The time derivative of V_1 is

$$\begin{aligned} \dot{V}_1 &= \frac{q(z_1) \xi_{b_1}^{2p-1}}{k_{b_1} (1 - \xi_{b_1}^{2p})} \left(z_2 + \sigma_1 - \dot{y}_d - z_1 \frac{\dot{k}_{b_1}}{k_{b_1}} \right) \\ &\quad + \frac{(1 - q(z_1)) \xi_{a_1}^{2p-1}}{k_{a_1} (1 - \xi_{a_1}^{2p})} \left(z_2 + \sigma_1 - \dot{y}_d - z_1 \frac{\dot{k}_{a_1}}{k_{a_1}} \right). \end{aligned} \quad (14)$$

Choose the virtual controller σ_1 as

$$\sigma_1 = -(\kappa_1 + \bar{\kappa}_1(t)) z_1 + \dot{y}_d - \frac{2p-1}{2p} z_1. \quad (15)$$

The time-varying gain is given as

$$\bar{\kappa}_i(t) = \sqrt{\left(\frac{\dot{k}_{a_i}}{k_{a_i}}\right)^2 + \left(\frac{\dot{k}_{b_i}}{k_{b_i}}\right)^2} + \beta_i, \quad (16)$$

where β_i and κ_i , $i = 1, 2$ are any positive constants. Make sure that the time derivative α_1 is bounded, when \dot{k}_{a_1} and \dot{k}_{b_1} are both zero. Substituting (15) and (16) into (14) and noting that

$$\bar{\kappa}_i + q(z_i) \frac{\dot{k}_{b_i}}{k_{b_i}} + (1 - q(z_i)) \frac{\dot{k}_{a_i}}{k_{a_i}} \geq 0 \quad (17)$$

we obtain

$$\begin{aligned} \dot{V}_1 &= \left(\frac{q(z_1) \xi_{b_1}^{2p-1}}{k_{b_1} (1 - \xi_{b_1}^{2p})} + \frac{(1 - q(z_1)) \xi_{a_1}^{2p-1}}{k_{a_1} (1 - \xi_{a_1}^{2p})} \right) z_2 \\ &\quad - z_1 \left(\bar{\kappa}_1(t) \frac{q(z_1) \xi_{b_1}^{2p}}{(1 - \xi_{b_1}^{2p})} + \bar{\kappa}_1(t) \frac{(1 - q(z_1)) \xi_{a_1}^{2p}}{(1 - \xi_{a_1}^{2p})} \right. \\ &\quad \left. + \frac{\dot{k}_{b_1}}{k_{b_1}} \frac{q(z_1) \xi_{b_1}^{2p}}{(1 - \xi_{b_1}^{2p})} + \frac{\dot{k}_{a_1}}{k_{a_1}} \frac{(1 - q(z_1)) \xi_{a_1}^{2p}}{(1 - \xi_{a_1}^{2p})} \right) \\ &\quad - \kappa_1 z_1 \left(\frac{q(z_1) \xi_{b_1}^{2p}}{(1 - \xi_{b_1}^{2p})} + \frac{(1 - q(z_1)) \xi_{a_1}^{2p}}{(1 - \xi_{a_1}^{2p})} \right) - \frac{2p-1}{2p} \\ &\quad \cdot \mu_1 z_1^{2p}, \end{aligned} \quad (18)$$

where

$$\mu_1 = \frac{q(z_1)}{k_{b_1}^{2p} - z_1^{2p}} + \frac{1 - q(z_1)}{k_{a_1}^{2p} - z_1^{2p}}. \quad (19)$$

After finishing it, we get

$$\begin{aligned} \dot{V}_1 = & \left(\frac{q(z_1)}{(k_{b_1}^{2p} - z_1^{2p})} + \frac{(1 - q(z_1))}{(k_{a_1}^{2p} - z_1^{2p})} \right) z_1^{2p-1} z_2 \\ & - \frac{\xi_1^{2p}}{(1 - \xi_1^{2p})} \left(\frac{\dot{k}_{b_1}}{k_{b_1}} q(z_1) + \frac{\dot{k}_{a_1}}{k_{a_1}} (1 - q(z_1)) \right. \\ & \left. + \bar{\kappa}_1(t) \right) - \frac{\kappa_1 \xi_1^{2p}}{(1 - \xi_1^{2p})} - \frac{2p-1}{2p} \mu_1 z_1^{2p}. \end{aligned} \quad (20)$$

Based on (12), we obtain

$$\dot{V}_1 \leq -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \mu_1 z_1^{2p-1} z_2 - \frac{2p-1}{2p} \mu_1 z_1^{2p}. \quad (21)$$

Using Young's inequality, the following inequality holds:

$$\mu_1 z_1^{2p-1} z_2 \leq \mu_1 \left(\frac{2p-1}{2p} z_1^{2p} + \frac{1}{2p} z_2^{2p} \right). \quad (22)$$

Substituting (22) into (21), \dot{V}_1 can be further rewritten as

$$\dot{V}_1 \leq -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \frac{1}{2p} \mu_1 z_2^{2p}. \quad (23)$$

The Barrier Lyapunov Function V_2 is given as

$$\begin{aligned} V_2 = & V_1 + \frac{q(z_2)}{2p} \log \frac{k_{b_2}^{2p}(t)}{k_{b_2}^{2p}(t) - z_2^{2p}} \\ & + \frac{1 - q(z_2)}{2p} \log \frac{k_{a_2}^{2p}(t)}{k_{a_2}^{2p}(t) - z_2^{2p}}. \end{aligned} \quad (24)$$

Then, differentiating of V_2 with respect to time is given by

$$\begin{aligned} \dot{V}_2 = & \dot{V}_1 + \frac{q(z_2) \xi_{b_2}^{2p-1}}{k_{b_2} (1 - \xi_{b_2}^{2p})} \left(\dot{z}_2 - z_2 \frac{\dot{k}_{b_2}}{k_{b_2}} \right) \\ & + \frac{(1 - q(z_2)) \xi_{a_2}^{2p-1}}{k_{a_2} (1 - \xi_{a_2}^{2p})} \left(\dot{z}_2 - z_2 \frac{\dot{k}_{a_2}}{k_{a_2}} \right). \end{aligned} \quad (25)$$

From the definition of the tracking error $z_2 = \alpha_2 - \sigma_1$, it is easy to obtain $\dot{z}_2 = \dot{\alpha}_2 - \dot{\sigma}_1$, and the derivative of the virtual controller is given as

$$\dot{\sigma}_1 = \frac{\partial \sigma_1}{\partial \alpha_1} \alpha_2 + \sum_{j=0}^1 \frac{\partial \sigma_1}{\partial \zeta_1} \zeta_1^{(j+1)}, \quad (26)$$

where $\zeta_1 = [y_d, k_{a_1}, k_{b_1}]^T$.

According to (26), (25) can be rewritten as

$$\begin{aligned} \dot{V}_2 = & (\dot{\alpha}_2 - \dot{\sigma}_1) \left(\frac{1 - q(z_2)}{k_{a_2}} \frac{\xi_{a_2}^{2p-1}}{1 - \xi_{a_2}^{2p}} + \frac{q(z_2)}{k_{b_2}} \frac{\xi_{b_2}^{2p-1}}{1 - \xi_{b_2}^{2p}} \right) \\ & - z_2 \frac{\dot{k}_{b_2}}{k_{b_2}} \frac{q(z_2) \xi_{b_2}^{2p-1}}{k_{b_2} (1 - \xi_{b_2}^{2p})} \\ & - z_2 \frac{\dot{k}_{a_2}}{k_{a_2}} \frac{1 - q(z_2) \xi_{a_2}^{2p-1}}{k_{a_2} (1 - \xi_{a_2}^{2p})} + \dot{V}_1. \end{aligned} \quad (27)$$

Based on (23), we get

$$\begin{aligned} \dot{V}_2 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \frac{1}{2p} \mu_1 z_2^{2p} + (\dot{\alpha}_2 - \dot{\sigma}_1) \frac{\xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} \\ & - z_2 \frac{\dot{k}_{b_2}}{k_{b_2}} \frac{q(z_2) \xi_{b_2}^{2p-1}}{k_{b_2} (1 - \xi_{b_2}^{2p})} \\ & - z_2 \frac{\dot{k}_{a_2}}{k_{a_2}} \frac{1 - q(z_2) \xi_{a_2}^{2p-1}}{k_{a_2} (1 - \xi_{a_2}^{2p})}. \end{aligned} \quad (28)$$

Substituting (26) into the above formula, we obtain

$$\begin{aligned} \dot{V}_2 = & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} \\ & + \frac{\xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} \left(J^{-1} (-f \dot{\alpha}_1 - T_f + u) - \dot{\sigma}_1 \right) \\ & - \frac{q(z_2) \xi_{b_2}^{2p-1}}{k_{b_2} (1 - \xi_{b_2}^{2p})} \left(z_2 \frac{\dot{k}_{b_2}}{k_{b_2}} \right) \\ & - \frac{1 - q(z_2) \xi_{a_2}^{2p-1}}{k_{a_2} (1 - \xi_{a_2}^{2p})} \left(z_2 \frac{\dot{k}_{a_2}}{k_{a_2}} \right) - \frac{J^{-1} \xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} d \\ & + \frac{1}{2p} \mu_1 z_2^{2p}. \end{aligned} \quad (29)$$

Using Young's inequality and noting $\|d(t)\| \leq d_M$, we obtain

$$-\frac{J^{-1} \xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} d \leq \frac{1}{2\gamma_1} \left(\frac{J^{-1} \xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} \right)^2 + \frac{1}{2} \gamma_1 d_M^2, \quad (30)$$

where γ_1 is a positive constant.

Based on (30), (29) can be rewritten as

$$\begin{aligned} \dot{V}_2 = & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \frac{1}{2p} \mu_1 z_2^{2p} + \frac{\xi_2^{2p}}{z_2 (1 - \xi_2^{2p})} \\ & \times (J^{-1}(-f\dot{\alpha}_1 - T_f + u) - \dot{\sigma}_1) + \frac{q(z_2)}{k_{b_2}} \frac{\xi_{b_2}^{2p-1}}{1 - \xi_{b_2}^{2p}} \\ & \times \left(-z_2 \frac{\dot{k}_{b_2}}{k_{b_2}} \right) + \frac{1 - q(z_2)}{k_{a_2}} \frac{\xi_{a_2}^{2p-1}}{1 - \xi_{a_2}^{2p}} \left(-z_2 \frac{\dot{k}_{a_2}}{k_{a_2}} \right) \\ & + \frac{1}{2\gamma_1} \left(\frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \right)^2 + \frac{1}{2}\gamma_1 d_M^2. \end{aligned} \quad (31)$$

For convenience, we define

$$M(Z) = J^{-1}(T_f + f\dot{\alpha}_1) + \dot{\sigma}_1. \quad (32)$$

In fact, since the parameters of f and T_f are not available, M is unknown in practice. In order to solve the uncertainty of this parameter, we designed NN, as shown below to estimate

$$M(Z) = W^{*T}S(Z) + \varepsilon(Z), \quad (33)$$

where $Z = [\alpha_1^T, \alpha_2^T]^T \in \Omega_z \subset R^3$, and similar to [28], we assume that the approximate error $\varepsilon(Z)$ satisfies $|\varepsilon(Z)| \leq \varepsilon^*$ with the constant $\varepsilon^* > 0$.

Substituting (33), (31) can be rewritten as

$$\begin{aligned} \dot{V}_2 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \frac{1}{2p} \mu_1 z_2^{2p} + \frac{J^{-1}\xi_2^{2p}\widehat{W}^T S(Z)}{z_2(1 - \xi_2^{2p})} \\ & - \frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \varepsilon(Z) + \frac{1}{2\gamma_1} \left(\frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \right)^2 \\ & + \frac{1}{2}\gamma_1 d_M^2 + \bar{\kappa}_2 \frac{\xi_2^{2p}}{1 - \xi_2^{2p}} + \frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} u. \end{aligned} \quad (34)$$

According to Young's inequality, we can easily obtain

$$-\frac{J^{-1}\xi_2^{2p}\varepsilon(Z)}{z_2(1 - \xi_2^{2p})} \leq \frac{1}{2\gamma_2} \left(\frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \right)^2 + \frac{1}{2}\gamma_2 \varepsilon^{*2}, \quad (35)$$

where γ_2 is a positive constant.

Based on (35), (34) can be rewritten as

$$\begin{aligned} \dot{V}_2 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} + \frac{1}{2p} \mu_1 z_2^{2p} + \frac{J^{-1}\xi_2^{2p}\widehat{W}^T S(Z)}{z_2(1 - \xi_2^{2p})} \\ & + \frac{1}{2\gamma_2} \left(\frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \right)^2 + \frac{1}{2}\gamma_2 \varepsilon^{*2} + \bar{\kappa}_2 \frac{\xi_2^{2p}}{1 - \xi_2^{2p}} \end{aligned}$$

$$\begin{aligned} & + \frac{1}{2\gamma_1} \left(\frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} \right)^2 + \frac{1}{2}\gamma_1 d_M^2 \\ & + \frac{J^{-1}\xi_2^{2p}}{z_2(1 - \xi_2^{2p})} u. \end{aligned} \quad (36)$$

The actual controller is given as

$$\begin{aligned} u = & -\frac{1}{2p} J \mu_1 z_2 (k_{b_2}^{2p} - z_2^{2p}) - \frac{J^{-1}\xi_2^{2p}}{2\gamma_1 z_2 (1 - \xi_2^{2p})} \\ & - \frac{J^{-1}\xi_2^{2p}}{2\gamma_2 z_2 (1 - \xi_2^{2p})} + J z_2 \widehat{W}^T S(Z) \\ & - J z_2 (\kappa_2 + \bar{\kappa}_2). \end{aligned} \quad (37)$$

Substituting (37), we obtain

$$\begin{aligned} \dot{V}_2 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} - \kappa_2 \frac{\xi_2^{2p}}{(1 - \xi_2^{2p})} + \frac{1}{2}\gamma_1 d_M^2 \\ & + \frac{\xi_2^{2p}}{1 - \xi_2^{2p}} \widehat{W}^T S(Z) + \frac{1}{2}\gamma_2 \varepsilon^{*2}. \end{aligned} \quad (38)$$

Design the Lyapunov function candidate V_3 :

$$V_3 = V_2 + \frac{1}{2} \widehat{W}^T \Gamma^{-1} \widehat{W}, \quad (39)$$

where $\Gamma = \Gamma^{-1} > 0$ is a constant matrix and $\widehat{W} = \widehat{W} - W^*$.

The time derivative of V_3 is

$$\dot{V}_3 = \dot{V}_2 + \widehat{W}^T \Gamma^{-1} \dot{\widehat{W}}. \quad (40)$$

Based on (38), we obtain

$$\begin{aligned} \dot{V}_3 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} - \kappa_2 \frac{\xi_2^{2p}}{(1 - \xi_2^{2p})} + \frac{1}{2}\gamma_1 d_M^2 + \frac{1}{2}\gamma_2 \varepsilon^{*2} \\ & + \widehat{W}^T \Gamma^{-1} \dot{\widehat{W}} + \frac{\xi_2^{2p}}{1 - \xi_2^{2p}} \widehat{W}^T S(Z). \end{aligned} \quad (41)$$

The adaptive law is given as follows:

$$\dot{\widehat{W}} = \Gamma \left(-\frac{\xi_2^{2p}}{1 - \xi_2^{2p}} S(Z) - \eta \widehat{W} \right), \quad (42)$$

where η is a positive constant.

Substituting (42) into (41), we get

$$\begin{aligned} \dot{V}_3 = & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} - \kappa_2 \frac{\xi_2^{2p}}{(1 - \xi_2^{2p})} + \frac{1}{2}\gamma_1 d_M^2 + \frac{1}{2}\gamma_2 \varepsilon^{*2} \\ & - \widehat{W}^T \eta (\widehat{W} + W^*). \end{aligned} \quad (43)$$

Using Young's inequality,

$$\begin{aligned} \dot{V}_3 \leq & -\frac{\kappa_1 \xi_1^{2p}}{1 - \xi_1^{2p}} - \kappa_2 \frac{\xi_2^{2p}}{(1 - \xi_2^{2p})} + \frac{1}{2} \gamma_1 d_M^2 + \frac{1}{2} \gamma_2 \varepsilon^{*2} \\ & - \frac{\eta}{2} \left(\|\widehat{W}\|^2 - \|W^*\|^2 \right). \end{aligned} \quad (44)$$

After finishing it, we get

$$\begin{aligned} \dot{V}_3 \leq & -\kappa_1 \log \frac{1}{1 - \xi_1^{2p}} - \frac{\eta}{2} \|\widehat{W}\|^2 + \frac{\eta}{2} \|W^*\|^2 \\ & - \kappa_2 \log \frac{1}{1 - \xi_2^{2p}} + \frac{1}{2} \gamma_1 d_M^2 + \frac{1}{2} \gamma_2 \varepsilon^{*2}. \end{aligned} \quad (45)$$

Then, the above inequality can be rewritten as

$$\dot{V}_3 \leq -\rho V_3 + C, \quad (46)$$

where

$$\begin{aligned} \rho &= \min \{ 2\kappa_1, 2\kappa_2, \eta \Gamma^{-1} \}. \\ C &= \frac{1}{2} \gamma_1 d_M^2 + \frac{1}{2} \gamma_2 \varepsilon^{*2} + \frac{\eta}{2} \|W^*\|^2 \end{aligned} \quad (47)$$

Theorem 7. Consider the unknown DC motor control system (1), based on the assumptions of Assumptions 2 and 3, Lemma 4, actual controller (37), and the adaptive law (42). The following properties guaranteed that the tracking error singles will remain in a compact neighborhood of zero, that is, $\lim_{t \rightarrow \infty} |y(t) - y_d(t)| = 0$, all signals of the closed-loop system are bounded, and all state constraints are never violated.

Proof. With both sides of (46) multiplied by $e^{\rho t}$, we obtain

$$\frac{d}{dt} \left(V_3 e^{\rho t} \right) \leq C e^{\rho t}. \quad (48)$$

Integrating (48) over $[0, t]$, we have

$$0 \leq V_3(t) \leq V_3(0) e^{-\rho t} + \frac{C}{\rho}. \quad (49)$$

Based on (7), (24), and (39), we can obtain

$$\begin{aligned} V_3 &= \frac{1}{2} \log \frac{k_{b_1}^{2p}(t)}{k_{b_1}^{2p}(t) - z_1^{2p}} + \frac{1}{2} \log \frac{k_{b_2}^{2p}(t)}{k_{b_2}^{2p}(t) - z_2^{2p}} \\ &+ \frac{1}{2} \widehat{W}^T \Gamma^{-1} \widehat{W}. \end{aligned} \quad (50)$$

Then, we have

$$\frac{1}{2} \log \frac{k_{b_1}^{2p}(t)}{k_{b_1}^{2p}(t) - z_1^{2p}} \leq V_{3A}(t) \leq V_{3A}(0), \quad (51)$$

where

$$\begin{aligned} V_{3A}(0) &= \frac{1}{2} \log \frac{k_{b_1}^{2p}(0)}{k_{b_1}^{2p}(0) - z_1^{2p}} \\ &+ \frac{1}{4} \lambda_{\max}(\Gamma^{-1}) \|\widehat{W}(0) - W^*\|^2 + \frac{C}{2\rho}. \end{aligned} \quad (52)$$

Therefore, we know that

$$\left(\frac{z_1}{k_b(t)} \right)^{2p} \leq 1 - e^{-2V_{3A}(0)}. \quad (53)$$

Based on the above inequality, the following inequality is obtained:

$$|z_1(t)| \leq D_1(t), \quad (54)$$

where

$$D_1(t) = k_{b_1}(t) \sqrt[2p]{1 - e^{-2V_{3A}(0)}}. \quad (55)$$

Similar to the derivation of z_1 , we can obtain the conclusion that

$$|z_2(t)| \leq D_2(t), \quad (56)$$

where

$$D_2(t) = k_{b_2}(t) \sqrt[2p]{1 - e^{-2V_{3B}(0)}}. \quad (57)$$

From Assumption 2, we can be known that $|\alpha_1(0)| < K_{c_1}(0)$, and from the definition of $k_{c_1}(t)$, we have $|z_1(t)| < k_{b_1}(t)$. In fact, from $\alpha_1 = z_1 + y_d$ and $\alpha_2 = z_2 + \sigma_1$, we obtain

$$|\alpha_1(t)| < k_{b_1}(t) + y_d(t). \quad (58)$$

Based on the above inequality, we know $|y(t)| \leq k_{c_1}(t)$, $\forall t \geq 0$. Therefore, the output signals are bounded.

Obviously, we can clearly obtain that the virtual controller σ_1 is bounded in (15). Based on $z_2 = \alpha_2 - \sigma_1$ and (56), α_2 is bounded. In addition, from (37) and (42), we know the actual controller u and the adaptive law \widehat{W} are bounded. Therefore, all the closed-loop system signals are bounded.

The proof is completed. \square

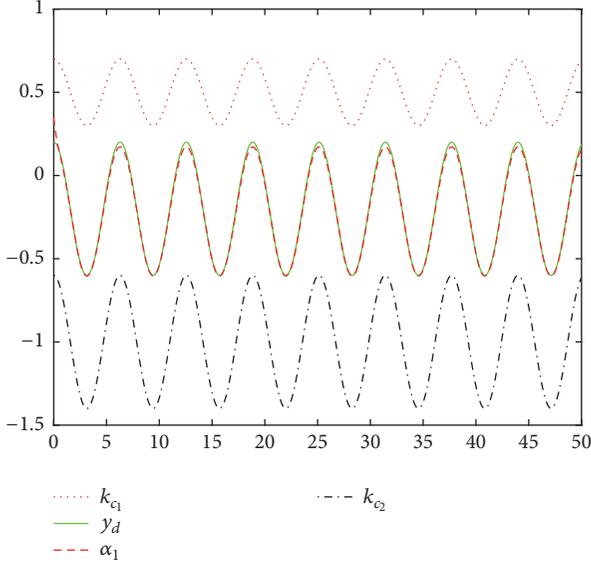
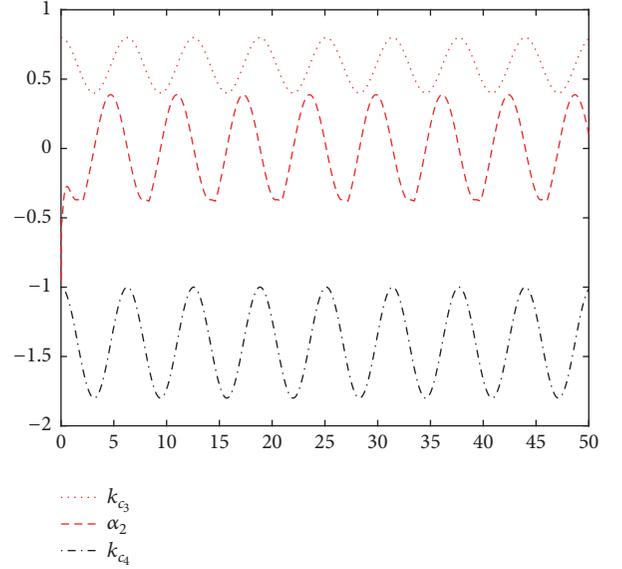
Remark 8. In the above analysis, it is apparent that the boundedness of z_1 lies on the design parameters γ_1 , γ_2 , η , d_M , ε^* , W^* , κ_1 , κ_2 , and Γ^{-1} . If we fix $\eta > 0$, it is clear that decreasing γ_i might result in small C and increasing κ_i might result in large ρ ; thus, it will help to reduce C/ρ . This represents that the tracking errors can be made arbitrarily small by selecting the design parameters appropriately.

4. Simulation Results

To illustrate the validity of the proposed adaptive NN control method, a simulation example is provided. Specifically, the following the DC motor system is described by

$$J \ddot{\alpha}_2 + f \dot{\alpha}_1 + T_f + d = u, \quad (59)$$

where the inertia is $J = 0.018 \text{ kg}\cdot\text{m}^2$, f denotes an unmeasured viscous friction with $f = \sin(t) + 1.82$, T_f is an unmeasured nonlinear friction with $T_f = 0.987$, and d is the external interference with $d = 0.05 \sin t$. The desired reference signal is given as $y_d = 0.4 \cos t - 0.2$. The virtual

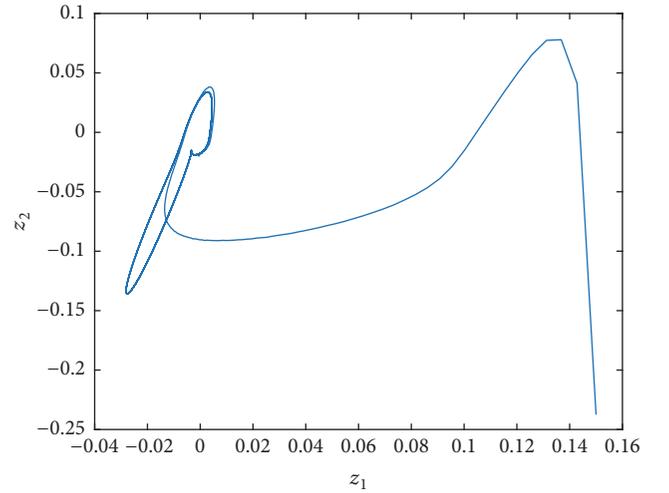
FIGURE 1: The trajectories of output α_1 and the reference signal $y_d(t)$.FIGURE 2: Trajectory of state α_2 .

controller, the actual controller, and the adaptation law are chosen as follows:

$$\begin{aligned}
 \sigma_1 &= -(\kappa_1 + \bar{\kappa}_1(t))z_1 + \dot{y}_d - \frac{2p-1}{2p}z_1 \\
 u &= -\frac{1}{2p}J\mu_1z_2(k_{b_2}^{2p} - z_2^{2p}) - J(\kappa_2 + \bar{\kappa}_2)z_2 \\
 &\quad - \frac{J^{-1}\xi_2^{2p}}{2\gamma_1z_2(1-\xi_2^{2p})} - \frac{J^{-1}\xi_2^{2p}}{2\gamma_2z_2(1-\xi_2^{2p})} \\
 &\quad + Jz_2\widehat{W}^T S(Z) \\
 \dot{\widehat{W}} &= \Gamma \left(-\frac{\xi_2^{2p}}{1-\xi_2^{2p}} S(Z) - \eta\widehat{W} \right).
 \end{aligned} \tag{60}$$

The angular position and the angular velocity of motor systems are bounded by $k_{c_1} < \alpha_1 < k_{c_2}$ and $k_{c_3} < \alpha_2 < k_{c_4}$ with $k_{c_1} = 0.5 + 0.2 \cos(t)$, $k_{c_2} = -1 + 0.4 \cos(t)$, $k_{c_3} = -1.4 + 0.4 \cos(t)$, and $k_{c_4} = 0.6 + 0.2 \cos(t)$. The NN $W^T S(z)$ contains 20 nodes and the centers μ_i , $i = 1, \dots, 20$. The design parameters of the proposed control method are chosen as $\Gamma = 2.5I$, $p = 2$, $\beta_1 = 4$, $\beta_2 = 4$, $\kappa_1 = 2$, $\kappa_2 = 2$, $c = 1$, and $T_f = 0.987$ and the initial condition of the system state is chosen as $\alpha_1(0) = 0.35$, $\alpha_2(0) = -0.95$, and $\widehat{W}(0) = 0$.

For the DC motor system, using a method of controlling the program can be obtained by the simulation results shown in Figures 1–5. Figures 1 and 2 show the output trajectory. Figure 1 shows the output and the reference signal tracking effect; the figure shows that the two curves almost coincide; that is to say, the tracking error converges to zero. Figure 3 shows the tracking error trajectory of $z_1(t)$ initially from the boundaries $k_{b_1}(t)$ and $-k_{a_1}(t)$ repulsion, but eventually converging to zero. Figure 4 shows a bounded and adaptive law of locus. According to Figure 4, we can see that the track

FIGURE 3: Phase portrait of z_1 and z_2 .

adaptation law is bounded. Thus, we can conclude that a good tracking performance can make all the signals in the closed-loop system bounded. From Figure 5, it can be observed that the control input is bounded by a bounded back and forth reciprocate.

5. Conclusion

In this paper, we propose an adaptive tracking control method for a DC system with full state constraints. The asymmetric time-varying BLF is employed to guarantee that the states always remain in the time-varying constrained sets. In the asymmetric system, neural networks and a backstepping technique are used to construct an adaptive control and adaptation laws to ensure that all signals in the closed-loop system are bounded and the state constraints are

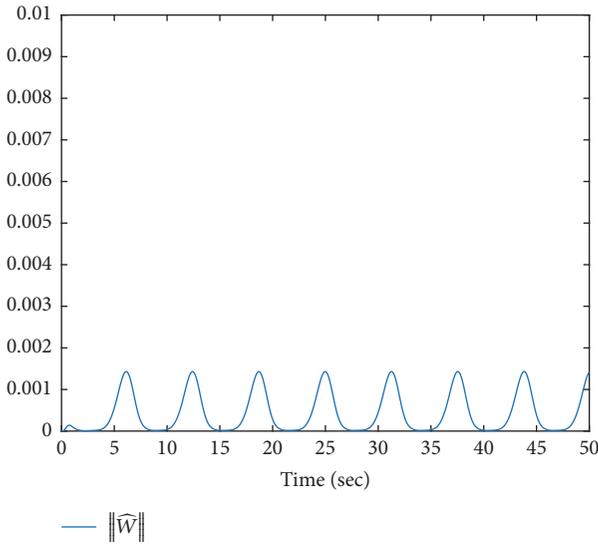


FIGURE 4: The trajectory of $\|\hat{W}\|$.

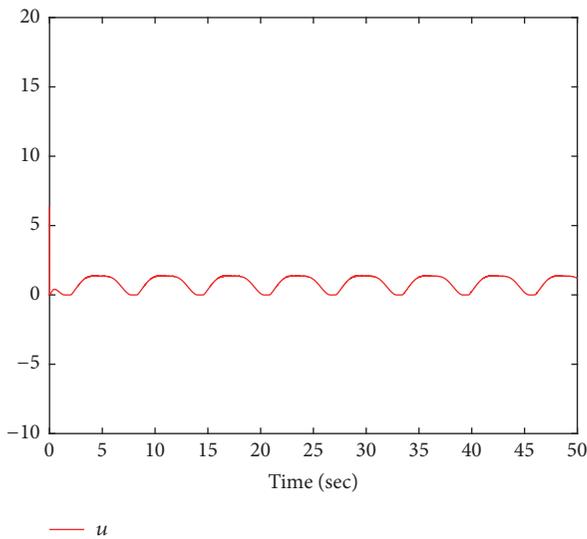


FIGURE 5: The trajectory of input u .

not transitioned. The performances of the adaptive control method based asymmetric time-varying BLF are verified by a simulation example.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61603164, 61473139, and 61622303) and the project for Distinguished Professor of Liaoning Province.

References

- [1] F. Deng, S. Guo, R. Zhou, and J. Chen, "Sensor Multifault Diagnosis with Improved Support Vector Machines," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 2, pp. 1053–1063, 2017.
- [2] F. Deng, S. Guan, X. Yue et al., "Energy-Based Sound Source Localization with Low Power Consumption in Wireless Sensor Networks," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 6, pp. 4894–4902, 2017.
- [3] L. Chen and Q. Wang, "Adaptive robust control for a class of uncertain MIMO non-affine nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 1, pp. 105–112, 2016.
- [4] M. Chen, "Robust tracking control for self-balancing mobile robots using disturbance observer," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 458–465, 2017.
- [5] P. Liu, Z. Zeng, and J. Wang, "Multiple Mittag-Leffler Stability of Fractional-Order Recurrent Neural Networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2279–2288, 2017.
- [6] Q. Shen, B. Jiang, and V. Cocquempot, "Fuzzy logic system-based adaptive fault-tolerant control for near-space vehicle attitude dynamics with actuator faults," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 2, pp. 289–300, 2013.
- [7] L. Liu, Z. Wang, and H. Zhang, "Adaptive fault-tolerant tracking control for MIMO discrete-time systems via reinforcement learning algorithm with less learning parameters," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 299–313, 2017.
- [8] Z. Li, H. Xiao, C. Yang, and Y. Zhao, "Model predictive control of nonholonomic chained systems using general projection neural networks optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 10, pp. 1313–1321, 2015.
- [9] J. Li, W. S. Chen, and J.-M. Li, "Adaptive neural network output-feedback decentralized stabilization for a class of large-scale stochastic nonlinear strict-feedback systems," *International Journal of Robust and Nonlinear Control*, vol. 21, no. 3, pp. 452–472, 2011.
- [10] W. S. Chen, L. C. Jiao, and Z. B. Du, "Output-feedback adaptive dynamic surface control of stochastic non-linear systems using neural network," *IET Control Theory & Applications*, vol. 4, no. 12, pp. 3012–3021, 2010.
- [11] Q. Zhou, H. Li, L. Wang, and R. Lu, "Prescribed Performance Observer-Based Adaptive Fuzzy Control for Nonstrict-Feedback Stochastic Nonlinear Systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [12] B. Xu, Z. Shi, C. Yang, and F. Sun, "Composite neural dynamic surface control of a class of uncertain nonlinear systems in strict-feedback form," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2626–2634, 2014.
- [13] W. Chen and L. Jiao, "Adaptive tracking for periodically time-varying and nonlinearly parameterized systems using multi-layer neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 21, no. 2, pp. 348–351, 2010.
- [14] T. Wang, H. Gao, and J. Qiu, "A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 416–425, 2016.
- [15] B. Chen, C. Lin, X. P. Liu, and K. F. Liu, "Observer-based adaptive fuzzy control for a class of nonlinear delayed systems,"

- IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 1, pp. 27–36, 2016.
- [16] L. Wang, M. Basin, H. Li, and R. Lu, “Observer-based Composite Adaptive Fuzzy Control for Nonstrict-feedback Systems with Actuator Failures,” *IEEE Transactions on Fuzzy Systems*, no. 99, pp. 1-1, 2017.
- [17] Z. Liu, C. Chen, Y. Zhang, and C. L. P. Chen, “Adaptive neural control for dual-arm coordination of humanoid robot with unknown nonlinearities in output mechanism,” *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 521–532, 2015.
- [18] A. Boulkroune, “A fuzzy adaptive control approach for nonlinear systems with unknown control gain sign,” *Neurocomputing*, vol. 179, pp. 318–325, 2016.
- [19] S. C. Tong, L. L. Zhang, and Y. M. Li, “Observed-based adaptive fuzzy decentralized tracking control for switched uncertain nonlinear large-scale systems with dead zones,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 1, pp. 37–47, 2016.
- [20] N. Shirkhani, M. A. Khanesar, and M. Teshnehlab, “Indirect model reference fuzzy control of SISO fractional order nonlinear chaotic systems,” *Procedia Computer Science*, vol. 102, pp. 309–316, 2016.
- [21] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, “Neural control of bimanual robots with guaranteed global stability and motion precision,” *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [22] S. S. Ge and C. Wang, “Adaptive neural control of uncertain MIMO nonlinear systems,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 15, no. 3, pp. 674–692, 2004.
- [23] S. C. Tong, S. Sui, and Y. Li, “Fuzzy adaptive output feedback control of MIMO nonlinear systems with partial tracking errors constrained,” *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 4, pp. 729–742, 2015.
- [24] D. P. Li and D. J. Li, “Adaptive neural tracking control for an uncertain state constrained robotic manipulator with time-varying delays,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [25] B. Niu, X. Zhao, X. Fan, and Y. Cheng, “A new control method for state-constrained nonlinear switched systems with application to chemical process,” *International Journal of Control*, vol. 88, no. 9, pp. 1693–1701, 2015.
- [26] M. U. Khan, S. Li, Q. Wang, and Z. Shao, “CPS Oriented Control Design for Networked Surveillance Robots with Multiple Physical Constraints,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 5, pp. 778–791, 2016.
- [27] W. He and S. S. Ge, “Cooperative control of a nonuniform gantry crane with constrained tension,” *Automatica*, vol. 66, pp. 146–154, 2016.
- [28] K. P. Tee, S. S. Ge, and E. H. Tay, “Barrier Lyapunov Functions for the control of output-constrained nonlinear systems,” *Automatica*, vol. 45, no. 4, pp. 918–927, 2009.
- [29] Z. Liu, G. Lai, Y. Zhang, and C. L. Chen, “Adaptive neural output feedback control of output-constrained nonlinear systems with unknown output nonlinearity,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 8, pp. 1789–1802, 2015.
- [30] H. Li, L. Bai, L. Wang, Q. Zhou, and H. Wang, “Adaptive neural control of uncertain nonstrict-feedback stochastic nonlinear systems with output constraint and unknown dead zone,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2048–2059, 2017.
- [31] Y.-J. Liu and S. Tong, “Barrier Lyapunov functions for Nussbaum gain adaptive control of full state constrained nonlinear systems,” *Automatica*, vol. 76, pp. 143–152, 2017.
- [32] D. P. Li and D. J. Li, “Adaptive neural tracking control for nonlinear time-delay systems with full state constraints,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1590–1601, 2017.
- [33] Y.-J. Liu, S. C. Tong, C. L. P. Chen, and D.-J. Li, “Adaptive NN control using integral barrier lyapunov functionals for uncertain nonlinear block-triangular constraint systems,” *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3747–3757, 2017.
- [34] X. Jin, “Adaptive fault tolerant control for a class of input and state constrained MIMO nonlinear systems,” *International Journal of Robust and Nonlinear Control*, vol. 26, no. 2, pp. 286–302, 2016.
- [35] K. P. Tee, B. Ren, and S. S. Ge, “Control of nonlinear systems with time-varying output constraints,” *Automatica*, vol. 47, no. 11, pp. 2511–2516, 2011.
- [36] Y. J. Liu, S. M. Lu, D. J. Li, and S. C. Tong, “Adaptive controller design-based ABLF for a class of nonlinear time-varying state constraint systems,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1546–1553, 2017.
- [37] C.-W. Hung, C.-T. Lin, C.-W. Liu, and J.-Y. Yen, “A variable-sampling controller for brushless DC motor drives with low-resolution position sensors,” *IEEE Transactions on Industrial Electronics*, vol. 54, no. 5, pp. 2846–2852, 2007.
- [38] J. Yao, Z. Jiao, and D. Ma, “Adaptive robust control of dc motors with extended state observer,” *IEEE Transactions on Industrial Electronics*, vol. 61, no. 7, pp. 3630–3637, 2014.
- [39] G. Yang, J. Yao, G. Le, and D. Ma, “Adaptive robust control of DC motors with time-varying output constraints,” in *Proceedings of the 34th Chinese Control Conference (CCC '15)*, pp. 4256–4261, China, July 2015.
- [40] L. Ma, S. Lu, M. Gong, J. Chen, and Y. Liu, “Time-varying output constraint for DC motor control,” in *Proceedings of the 3rd International Conference on Informative and Cybernetics for Computational Social Systems (ICCSS '16)*, pp. 151–154, China, August 2016.

Research Article

Application of Federal Kalman Filter with Neural Networks in the Velocity and Attitude Matching of Transfer Alignment

Lijun Song , Zhongxing Duan , Bo He, and Zhe Li

Electronic Information and Control Engineering College, Xi'an University of Architecture and Technology, Xi'an 710055, China

Correspondence should be addressed to Zhongxing Duan; zhx_duan@163.com

Received 8 July 2017; Accepted 8 October 2017; Published 21 January 2018

Academic Editor: Junpei Zhong

Copyright © 2018 Lijun Song et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The centralized Kalman filter is always applied in the velocity and attitude matching of Transfer Alignment (TA). But the centralized Kalman has many disadvantages, such as large amount of calculation, poor real-time performance, and low reliability. In the paper, the federal Kalman filter (FKF) based on neural networks is used in the velocity and attitude matching of TA, the Kalman filter is adjusted by the neural networks in the two subfilters, the federal filter is used to fuse the information of the two subfilters, and the global suboptimal state estimation is obtained. The result of simulation shows that the federal Kalman filter based on neural networks is better in estimating the initial attitude misalignment angle of inertial navigation system (INS) when the system dynamic model and noise statistics characteristics of inertial navigation system are unclear, and the estimation error is smaller and the accuracy is higher.

1. Introduction

Because there are too many vector dimensions in TA, the centralized Kalman filter has many disadvantages, such as large amount of calculation, poor real-time performance, and low reliability. In theory, the federal filtering is optimal or suboptimal, and it has the characteristics of high reliability, design flexibility, and being easy to apply to data fusion of navigation. In the engineering practice, system noise and measurement noise are always random signal, so it is difficult to get the statistic characteristics. The federal Kalman filter with neural networks is proposed in the paper. The federal Kalman filter is combined with the neural networks to improve the precision of initial attitude misalignment angle of inertial navigation system [1, 2].

2. The Theory of Federal Kalman Filter with Neural Networks

2.1. The Traditional Kalman Filter. It is assumed that the system equation and measurement equation of linear discrete system can be written as follows:

$$\begin{aligned} \mathbf{X}_k &= \Phi_{k,k-1} \mathbf{X}_{k-1} + \mathbf{W}_{k-1} \\ \mathbf{Z}_k &= \mathbf{H}_k \mathbf{X}_k + \mathbf{V}_k, \end{aligned} \quad (1)$$

where \mathbf{X}_k is estimated state, \mathbf{Z}_k is the measurement of system, and $\Phi_{k,k-1}$ is transfer matrix from t_{k-1} to t_k . \mathbf{W}_{k-1} is noise sequence of system incentive. \mathbf{H}_k is the measurement matrix, \mathbf{V}_k is noise sequence of measurement, and $E[\mathbf{V}_k \mathbf{V}_k^T] = \mathbf{R}$.

So the Kalman filter equation is shown as follows.

(1) Time update is

$$\begin{aligned} \hat{\mathbf{X}}_{k/k-1} &= \Phi_{k,k-1} \hat{\mathbf{X}}_{k-1} \\ \mathbf{P}_{k/k-1} &= \Phi_{k,k-1} \mathbf{P}_{k-1} \Phi_{k,k-1}^T + \Gamma_{k-1} \mathbf{Q}_{k-1} \Gamma_{k-1}^T. \end{aligned} \quad (2)$$

(2) Measurement update is

$$\begin{aligned} \mathbf{K}_k &= \mathbf{P}_{k/k-1} \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_{k/k-1} \mathbf{H}_k^T + \mathbf{R}_k)^{-1} \\ \hat{\mathbf{X}}_k &= \hat{\mathbf{X}}_{k/k-1} + \mathbf{K}_k (\mathbf{Z}_k - \mathbf{H}_k \hat{\mathbf{X}}_{k/k-1}) \\ \mathbf{P}_k &= (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k/k-1} (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)^T + \mathbf{K}_k \mathbf{R}_k \mathbf{K}_k^T. \end{aligned} \quad (3)$$

The state estimate \mathbf{X}_k could be calculated when the initial values \mathbf{X}_0 and \mathbf{P}_0 are known.

2.2. The Federal Filter. The federal filter is composed of several subfilters and a main filter, where the subfilter can

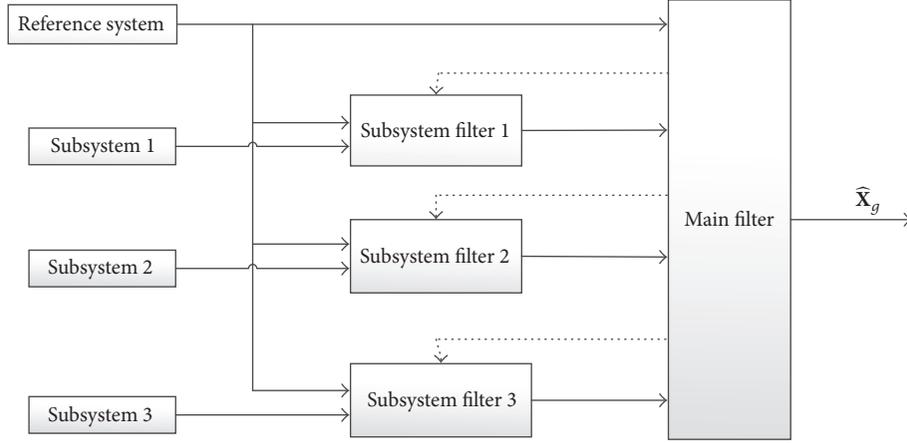


FIGURE 1: The structure of the federal filter.

independently update the time and measurement, and the result will be sent to the main filter. And the main filter will feed back to the subfilter after data fusion is analyzed. It is the initial value of the next cycle. The federal filter is that the large matrix of centralized filter algorithm is divided into the subfilters [3, 4]. The structure of the federal filter is shown in Figure 1.

The fusion algorithm of the traditional federal filter is as follows:

$$\begin{aligned} \hat{\mathbf{X}}_g &= \mathbf{P}_g \left(\sum_{i=1}^m \mathbf{P}_{ci}^{-1} \hat{\mathbf{X}}_{ci} \right) \\ \mathbf{P}_g &= \left(\sum_{i=1}^m \mathbf{P}_{ci}^{-1} \right)^{-1}, \end{aligned} \quad (4)$$

where $\hat{\mathbf{X}}_g$ is global state estimation after fusion, $\hat{\mathbf{X}}_{ci}$ is the local optimum estimation of subfilter on the system of public state, \mathbf{P}_{ci} is variance matrix of error of the local optimum estimation, and the local estimations are not related to each other.

2.3. The BP Neural Networks. The uptime of Kalman filter is proportional to the cubic of the system order. The Kalman filter will lose the real-time performance when the order of system is very high, so the precision is not improved by the order of system.

Artificial neural network is a system that simulates the structure and function of human brain neuron networks by method of engineering technology. It is comprised of a large number of simple nonlinear processing units, and there are complex and flexible connections between units.

The neural networks have a strong self-learning ability and arbitrary nonlinear functions and parallel processing could be realized. So the neural networks are always applied in the information technology and control engineering. According to the characteristics of information transmission, neural networks can be divided into feedforward and feedback. The BP neural networks are feedforward networks which are based on error backpropagation algorithm [5–7].

There are input layer nodes, one or more hidden-layer nodes, and output layer nodes in the BP neural networks.

If the number of hidden-layers and nodes in each layer are determined, the structure of the BP neural networks is determined [8–10]. The principle of BP neural networks is shown as Figure 2.

2.4. The Federal Kalman Filter with Neural Networks

2.4.1. The Theory of Federal Kalman Filter. The theory of federal Kalman filters in TA is as follows:

- (1) The output date of master inertial navigation system (MINS) is used as the common reference system, the velocity and angularity of slaver inertial navigation system (SINS) regarded as independent subsystems. In order to solve the problem, the BP neural networks are used in the two subsystems while the statistics characteristics of the system dynamic model and noise model are unclear.
- (2) The output date of two subsystems used the federal filter to obtain global suboptimal estimation.

2.4.2. The Theory of Kalman Filter with Neural Networks. The purpose of the subfilter which used the BP neural networks is that the Kalman filter is followed by the BP neural networks with the minimum error.

The training process of neural networks is as follows: first, the Kalman filter worked as open loop state to eliminate the influence of convergence rate for initial value. Second, the Kalman filter worked as closed loop state, and the initial sample values of the neural network are constituted by the input and output of the Kalman filter. Lastly, the sample value of training could be got by averaging the testing values of measurements. After the training, The BP neural networks will be worked alone when the Kalman filter is removed [11–13].

The inputs of BP neural networks are comprised of three parts: ① the difference of state prediction and state estimation is $\hat{\mathbf{X}}_k - \hat{\mathbf{X}}_{k/k-1}$. ② The difference of observation and observation estimation is $\mathbf{Z}_k - \mathbf{H}_k \hat{\mathbf{X}}_{k/k-1}$. ③ The filter gain is \mathbf{K}_k . $\Delta \hat{\mathbf{X}}_k = \mathbf{D} - \hat{\mathbf{X}}_k$ is treated as the desired output of BP neural networks, where \mathbf{D} is the theoretical value of state vector. $\Delta \hat{\mathbf{X}}_k$

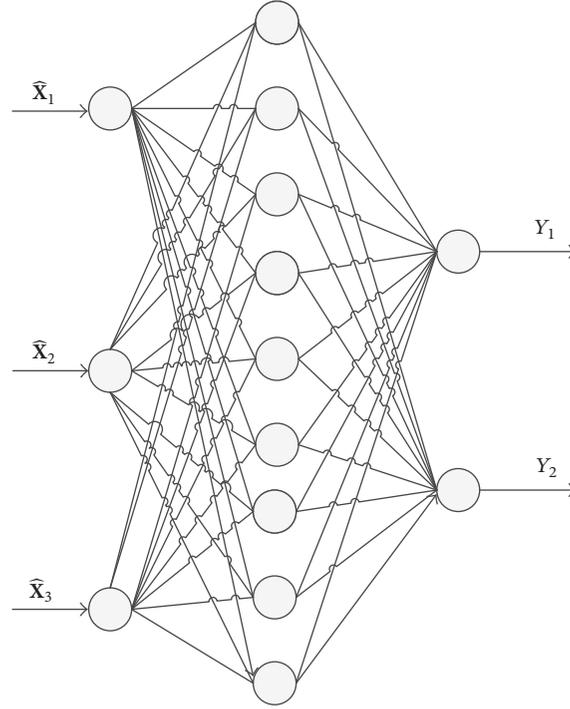


FIGURE 2: The principle of BP neural networks.

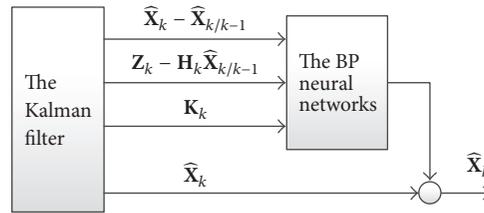


FIGURE 3: The principle of Kalman filters with neural networks.

is the output of BP neural networks and $\widehat{\mathbf{X}}_k$ is the output of Kalman filter, where the two outputs are added together after the BP neural networks are trained by samples.

The principle of Kalman filters with neural networks is shown as Figure 3.

3. The Velocity and Attitude Matching in TA

The velocity matching in TA is required to assist maneuver (such as right circle), and attitude matching in TA is required angular velocity of pitch or roll (such as wing maneuver), but it is not separated from the head platform misalignment angle in the velocity matching and the same in the attitude matching when the plane has the wing maneuver [13, 14].

In the velocity and attitude matching, horizontal alignment is realized in the velocity and azimuth alignment is realized in attitude matching. The wing maneuver is required during the alignment procedure.

3.1. State Equation of the Velocity and Attitude Matching. It is supposed that $\mathbf{X} = [\boldsymbol{\varphi}^{nT} \ \delta\mathbf{V}_e^{nT} \ \boldsymbol{\varepsilon}^{b_sT} \ \nabla^{b_sT} \ \boldsymbol{\mu}^T \ \boldsymbol{\lambda}_f^T \ \boldsymbol{\omega}_f^T]^T$ is the system state equation of the velocity and attitude

matching, where $\boldsymbol{\varphi}^n = [\varphi_x \ \varphi_y \ \varphi_z]^T$ is the platform misalignment angle of slaver inertial navigation system, $\delta\mathbf{V}_e^n = [\delta V_{ex}^n \ \delta V_{ey}^n \ \delta V_{ez}^n]^T$ is velocity error of slaver inertial navigation system, $\boldsymbol{\mu} = [\mu_x \ \mu_y \ \mu_z]^T$ is error of missile body's installation angle, $\nabla^{b_s} = [\nabla_x^{b_s} \ \nabla_y^{b_s} \ \nabla_z^{b_s}]^T$ is accelerometer's constant error of slaver inertial navigation system, $\boldsymbol{\lambda}_f = [\lambda_{fx} \ \lambda_{fy} \ \lambda_{fz}]^T$ is flexure deformation angle of wing, and $\boldsymbol{\omega}_f = [\omega_{fx} \ \omega_{fy} \ \omega_{fz}]^T$ is flexure deformation angular rate of wing. State equation of the velocity and attitude matching is

$$\begin{aligned} \dot{\boldsymbol{\varphi}}^n &= -\boldsymbol{\omega}_{in}^n \times \boldsymbol{\varphi}^n - \mathbf{C}_{b_s}^n \boldsymbol{\varepsilon}_b^{b_s} - \mathbf{C}_{b_s}^n \boldsymbol{\varepsilon}_w^{b_s} \\ \delta\dot{\mathbf{V}}^n &= (\mathbf{C}_{b_s}^n \mathbf{f}^{b_s}) \times \boldsymbol{\varphi} - (2\boldsymbol{\omega}_{ie}^n + \boldsymbol{\omega}_{en}^n) \times \delta\mathbf{V}^n + \mathbf{C}_{b_s}^n \nabla_b^{b_s} \\ &\quad + \mathbf{C}_{b_s}^n \nabla_w^{b_s} \\ \dot{\boldsymbol{\varepsilon}}^{b_s} &= \mathbf{0} \\ \dot{\nabla}^{b_s} &= \mathbf{0} \\ \dot{\boldsymbol{\mu}}^{b_f} &= \mathbf{0} \end{aligned}$$

$$\begin{aligned}\dot{\lambda}_f &= \omega_f, \\ \dot{\omega}_f &= -[\beta^2] \lambda_f - [\beta] \omega_f + \eta\end{aligned}\quad (5)$$

So the state-space model of the velocity and attitude matching is

$$\dot{\mathbf{X}} = \begin{bmatrix} -(\omega_{in}^n \times) & \mathbf{0}_{3 \times 3} & -\mathbf{C}_{b_s}^n & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ (\mathbf{C}_{b_s}^n \mathbf{f}^{b_s} \times) - ((2\omega_{ie}^n + \omega_{en}^n) \times) & \mathbf{0}_{3 \times 3} & \mathbf{C}_{b_s}^n & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & I_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & -[\beta^2] & -[\beta] \end{bmatrix} \mathbf{X} + \begin{bmatrix} -\mathbf{C}_{b_s}^n \boldsymbol{\varepsilon}_w^{b_s} \\ \mathbf{C}_{b_s}^n \nabla_w^{b_s} \\ \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} \\ \boldsymbol{\eta} \end{bmatrix}, \quad (6)$$

where $\mathbf{C}_{b_s}^n = \begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix}$ is the attitude matrix of slaver inertial navigation system:

$$\begin{aligned}(\mathbf{C}_{b_s}^n \mathbf{f}^{b_s} \times) &= \begin{bmatrix} 0 & -f_U^n & f_N^n \\ f_U^n & 0 & -f_E^n \\ -f_N^n & f_E^n & 0 \end{bmatrix} \\ ((2\omega_{ie}^n + \omega_{en}^n) \times) &= \begin{bmatrix} 0 & -\left(2\omega_{ie} \sin L + \frac{V_E \tan L}{R_N}\right) & \left(2\omega_{ie} \cos L + \frac{V_E}{R_N}\right) \\ \left(2\omega_{ie} \sin L + \frac{V_E \tan L}{R_N}\right) & 0 & \frac{V_N}{R_M} \\ -\left(2\omega_{ie} \cos L + \frac{V_E}{R_N}\right) & \frac{-V_N}{R_M} & 0 \end{bmatrix}, \quad (7)\end{aligned}$$

where $\boldsymbol{\varepsilon}_w^{b_s}$ is Gaussian white noise of gyro, $\nabla_w^{b_s}$ is Gaussian white noise of accelerometer, $\boldsymbol{\eta} = [\eta_x \ \eta_y \ \eta_z]^T$ is Gaussian white noise sequence of second order, in which $\eta_i \sim \mathbf{N}(0, Q_i)$, $Q_i = 4\beta_i^3 \sigma_\eta^2$, and σ_η^2 is the variance of flexure deformation angle; $[\beta] = \text{diag}(\beta_x, \beta_y, \beta_z)$ and $[\beta^2] = \text{diag}(\beta_x^2, \beta_y^2, \beta_z^2)$.

3.2. The Measurement Equation of Velocity and Attitude Matching. It is supposed that $\widehat{\mathbf{V}}_{em}^n$ is the velocity of MINS, $\widehat{\mathbf{V}}_{es}^n$ is the velocity of SINS, $\widehat{\mathbf{V}}_{LA}^n$ is the lever velocity from MINS, $\widehat{\mathbf{C}}_{b_m}^n$ is the attitude matrix of MINS, $\widehat{\mathbf{C}}_{b_s}^n$ is the attitude matrix of SINS, and $\mathbf{C}_{b_f}^{b_h}$ is transformation matrix between b_f and b_h , where b_f is installation coordinate and b_h is horizontal coordinate.

The difference of velocity between MINS and SINS is the measurement in the velocity matching; attitude matrix is the measurement in the attitude matching. The measurement of the velocity and attitude matching is as follows.

Consider

$$\mathbf{Z}_V = \widehat{\mathbf{V}}_{es}^n - (\widehat{\mathbf{V}}_{em}^n + \widehat{\mathbf{V}}_{LA}^n)$$

$$\mathbf{Z} = \begin{bmatrix} Z_V \\ Z_\theta \end{bmatrix}$$

$$\mathbf{Z}_\theta = \begin{bmatrix} \frac{\mathbf{Z}_{DCM}(3,2) - \mathbf{Z}_{DCM}(2,3)}{2} \\ \frac{\mathbf{Z}_{DCM}(1,3) - \mathbf{Z}_{DCM}(3,1)}{2} \\ \frac{\mathbf{Z}_{DCM}(2,1) - \mathbf{Z}_{DCM}(1,2)}{2} \end{bmatrix}$$

$$\begin{aligned}\mathbf{Z}_{DCM} &= \widehat{\mathbf{C}}_{b_m}^n \mathbf{C}_{b_f}^{b_h} \widehat{\mathbf{C}}_{b_s}^n \\ &= [\mathbf{I} - (\boldsymbol{\varphi}_m^n \times)] \mathbf{C}_{b_m}^n \mathbf{C}_{b_f}^{b_h} \mathbf{C}_{b_s}^n [\mathbf{I} + (\boldsymbol{\varphi}^n \times)].\end{aligned}\quad (8)$$

$\boldsymbol{\varphi}_m^n$ is the attitude error angle of master inertial navigation system, and $\boldsymbol{\varphi}^n$ is the attitude error angle of slaver inertial navigation system.

The measurement equation of velocity and attitude matching is

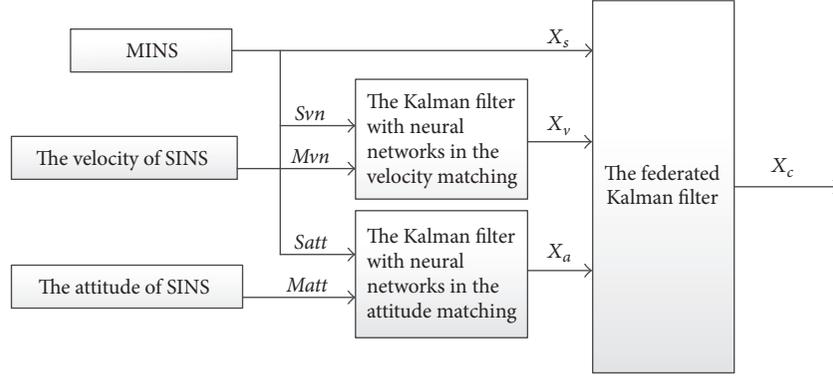


FIGURE 4: The principle diagram of the federated Kalman filter with neural networks.

$$\mathbf{Z} = \begin{bmatrix} \mathbf{0}_{3 \times 3} & \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} \\ \mathbf{I}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & \mathbf{0}_{3 \times 3} & -\mathbf{C}_{b_m}^n & \mathbf{C}_{b_f}^{b_h} & -\mathbf{C}_{b_m}^n & \mathbf{0}_{3 \times 3} \end{bmatrix} \mathbf{X} + \begin{bmatrix} \mathbf{V}_V \\ \mathbf{V}_\theta \end{bmatrix}, \quad (9)$$

where the white noise of zero mean Gaussian is \mathbf{V}_V , and the measurement noise signal of unknown is \mathbf{V}_θ .

4. Application of Federal Kalman Filter with Neural Networks in the Velocity and Attitude Matching

4.1. The Structure Design of Federal Kalman Filter with Neural Networks. The centralized Kalman filter is applied in the velocity and attitude matching of TA. It is superior to the velocity matching or the attitude matching, especially in the speed of alignment and the requirement of maneuver, but there are too many numbers of dimensions and large amount of calculation. Combined with the neural networks, the performance of federal Kalman filter is improved on the velocity and attitude matching when the statistics characteristics of the system dynamic model and noise model are unclear [15, 16].

The parallel computation is supported by the federal Kalman filter with neural networks to reduce the calculation in TA, and the neural networks are used in the subfilter. The principle diagram of the federal Kalman filter with neural networks on the velocity and attitude matching is shown as Figure 4.

4.2. The Simulation of Federal Kalman Filter with Neural Networks in the Velocity and Attitude Matching. It is supposed that there are the flexure deformation of the wing and the unknown measurement noise signal when the federal Kalman filters with neural networks are simulated in the velocity and attitude matching.

According to the flight characteristic, this paper makes the corresponding numerical simulation under wing motion by aircraft. The time of shake wing is 60 s and the angle of shake

wing is 30° . The initial position of TA is that north latitude is 34.03006° , east longitude is 108.76405° , and altitude is 448 m; the initial attitude of TA is that yaw is -90° , roll is 0° , and pitch is 0° .

The condition for simulation is as follows: error parameters of SINS: constant drift of gyro is $1^\circ/\text{h}$, random walk of gyro is $0.1^\circ/\sqrt{\text{h}}$, constant offset error of accelerometer is $5 \times 10^{-4} \text{ g}$, and standard deviation of accelerometer is $5 \times 10^{-5} \text{ g} \cdot \sqrt{\text{s}}$.

Installing-error angle of missile body: $\boldsymbol{\mu} = [0.1^\circ \ 0.1^\circ \ 0.1^\circ]^T$.

Misalignment initial angle of SINS: $\boldsymbol{\varphi}(0) = [0.1^\circ \ 0.1^\circ \ 0.5^\circ]^T$.

Velocity initial error of SINS: $\delta \mathbf{V}_e^n(0) = [3 \text{ m/s} \ 3 \text{ m/s} \ 3 \text{ m/s}]^T$.

The number of input layer nodes of BP neural networks is 6, the number of hidden-layer nodes of BP neural networks is 12, and the number of output layer nodes of BP neural networks is 3.

The simulation period is 20 ms. Meanwhile, it has the same conditions as Kalman filter. The blue solid line is the result of the federal Kalman filter with neural networks, and the red dashed line is the result of Kalman filter.

It is the main difference between the federal Kalman filter with neural networks and the Kalman filter that the Kalman filters with neural networks are used in the velocity matching and the attitude matching, and the result is got by the Kalman filter with neural networks in the velocity matching and the attitude matching. In order to get the optimal estimation of navigation parameter error, the result is sent to the main filter for data fusion.

From Figure 5, it is showed that the convergence of misalignment angle of the Kalman filter has been in $10'$ after 20 s. But the convergence of misalignment angle of the federal Kalman filter with neural networks has been in $5'$ after 20 s. The precision of federal Kalman filter with neural networks is above the precision of Kalman filter.

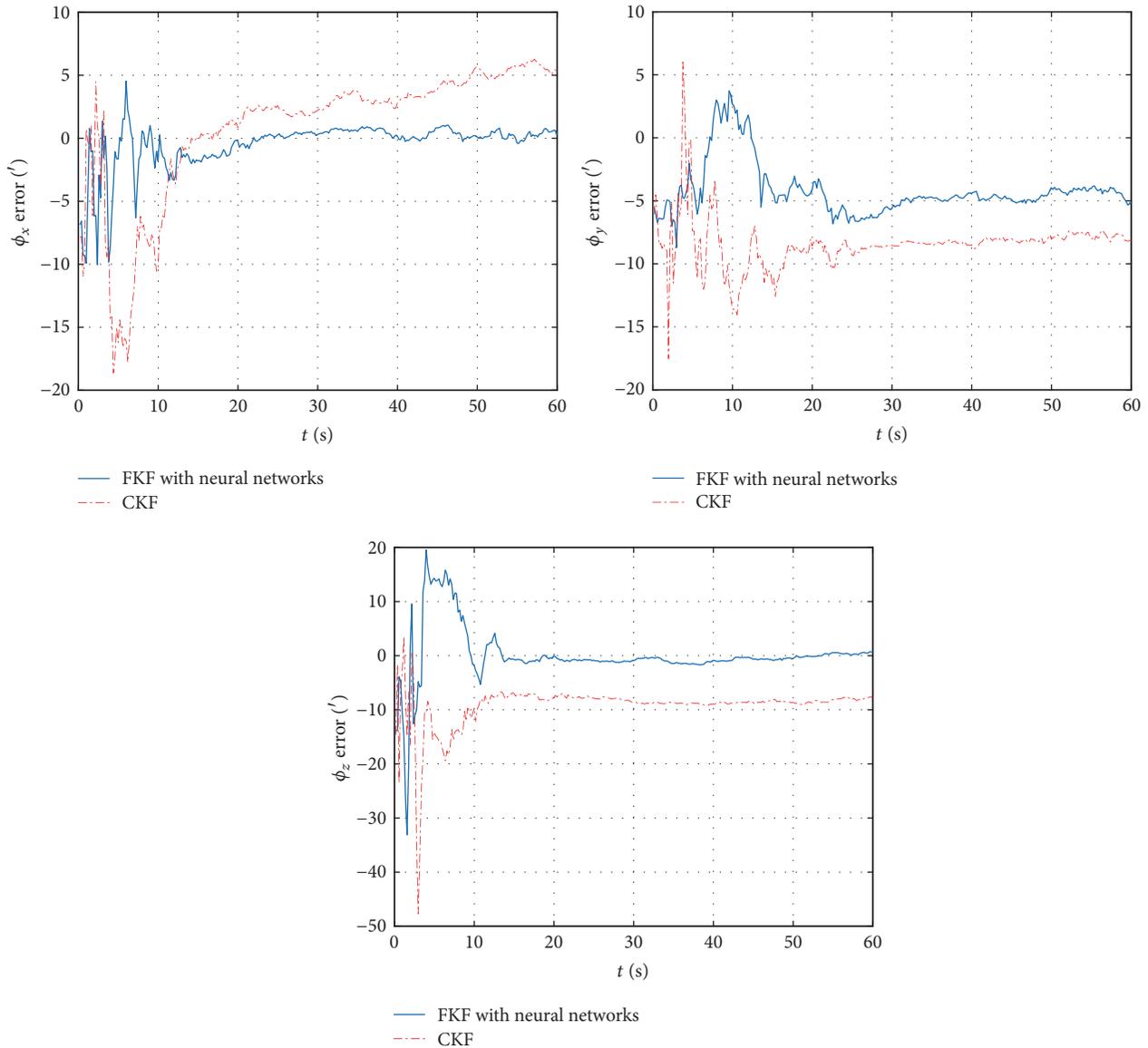


FIGURE 5: Estimate error of misalignment angle.

5. Summary

The input of the networks is the error parameters of Kalman filter. After the sample training, the BP networks output is added to the results of Kalman filter. In the practical engineering applications, this algorithm is superior to the Kalman filter under the nonlinear situation, and the result of simulation shows that the federal Kalman filter with neural networks in the velocity and attitude matching is more practicable and effective. The federal Kalman filter with neural networks is better to estimate the initial attitude misalignment angle of inertial navigation system when the system dynamic model and noise statistics characteristics of inertial navigation system are unclear, and the estimation error is smaller and the accuracy is higher.

Conflicts of Interest

The authors declare that they do not have any commercial or associative interest that represents conflicts of interest in connection with the work submitted.

Acknowledgments

This work is partially supported by the special scientific research project of the Education Department of Shaanxi Provincial Government, School Foundation Research Fund, Natural Science Foundation of Shaanxi Provincial Department and the National Natural Science Foundation of China, the project Number: 15JK1416, JC1702, 2016JM6084 and 51678470.

References

- [1] X.-J. Guan and X.-L. Wang, "Transfer alignment match methods for strapdown inertial navigation system on moving bases," *Aero Weapons*, vol. 4, no. 2, pp. 3–8, 2014.
- [2] G.-l. Yang, L.-f. Wang, E.-k. Yuan, L. Cai, and L.-w. Qiao, "Rapid transfer alignment for carrier - based aircrafts in catapult," in *Ship Science and Technol*, vol. 36, pp. 50–54, 3 edition, 2014.
- [3] L. Zhang, S. Qian, S. Zhang, and H. Cai, "Federated nonlinear predictive filtering for the gyroless attitude determination system," *Advances in Space Research*, vol. 58, no. 9, pp. 1671–1681, 2016.
- [4] Q. Y. yuan and N. I. Huifang, "Application of federated filtering theory to designing integrated navigation system," *Journal of Chinese Inertial Technology*, vol. 5, no. 3, pp. 1–5, 1997.
- [5] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, 2017.
- [6] L. Dan, "Improved BP neural network WSN data fusion scheme," *Information Technology*, vol. 37, no. 2, pp. 155–160, 2017.
- [7] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Y. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.
- [8] M. Yongjun, X. Yonghao, L. Yang, and L. Yajun, "Data aggregation algorithm based on the model of deep learning," *Journal of Tianjin University of Science Technology*, vol. 32, no. 4, pp. 1–5, 2017.
- [9] C. Yang, K. Huang, H. Cheng, Y. Li, and C.-Y. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.
- [10] T. Zengshan and C. Yongquan, "Attitude measurement fusion algorithm in GPS/SINS based on BP neural-network," *Journal of Chongqing University of Posts and Telecommunications*, vol. 26, no. 4, pp. 478–482, 2014.
- [11] X. Tianlai and M. Xu, "INS/GPS integrated navigation method based on elman neural network," in *Proceedings of the International Conference on Fuzzy Systems and Neural Computing*, pp. 310–313, 2011.
- [12] B. Feng, H. Ma, M. Fu, and C. Yang, "Real-time state estimator without noise covariance matrices knowledge-fast minimum norm filtering algorithm," *IET Control Theory & Applications*, vol. 9, no. 9, pp. 1422–1432, 2015.
- [13] G.-M. Yan, J. Weng, P.-X. Yang, and Y.-y. Qin, "Study on SINS rapid gyrocompass initial alignment," *International Symposium on Inertial Technology and Navigation*, pp. 323–330, 2010.
- [14] K. Kim, S. Seol, and S.-H. Kong, "High-speed train navigation system based on multi-sensor data fusion and map matching algorithm," *International Journal of Control, Automation, and Systems*, vol. 13, no. 3, pp. 503–512, 2015.
- [15] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2125–2136, 2017.
- [16] L. Bin, *The research and design of transfer alignment simulation and verification system on ship carried weapon inertial navigation system*, Harbin Engineering University, Harbin, China, 2012.

Research Article

Gender and Handedness Prediction from Offline Handwriting Using Convolutional Neural Networks

Ángel Morera, Ángel Sánchez , José Francisco Vélez, and Ana Belén Moreno

Technical School of Computer Science, Rey Juan Carlos University, Móstoles, 28933 Madrid, Spain

Correspondence should be addressed to Ángel Sánchez; angel.sanchez@urjc.es

Received 21 July 2017; Revised 5 October 2017; Accepted 12 November 2017; Published 14 January 2018

Academic Editor: Jing Na

Copyright © 2018 Ángel Morera et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Demographic handwriting-based classification problems, such as gender and handedness categorizations, present interesting applications in disciplines like Forensic Biometrics. This work describes an experimental study on the suitability of deep neural networks to three automatic demographic problems: gender, handedness, and combined gender-and-handedness classifications, respectively. Our research was carried out on two public handwriting databases: the IAM dataset containing English texts and the KHATT one with Arabic texts. The considered problems present a high intrinsic difficulty when extracting specific relevant features for discriminating the involved subclasses. Our solution is based on convolutional neural networks since these models had proven better capabilities to extract good features when compared to hand-crafted ones. Our work also describes the first approach to the combined gender-and-handedness prediction, which has not been addressed before by other researchers. Moreover, the proposed solutions have been designed using a unique network configuration for the three considered demographic problems, which has the advantage of simplifying the design complexity and debugging of these deep architectures when handling related handwriting problems. Finally, the comparison of achieved results to those presented in related works revealed the best average accuracy in the gender classification problem for the considered datasets.

1. Introduction

In spite of current technological advances, there are not still algorithms allowing a computer to transcribe the content of any “difficult” handwritten document (e.g., a historical document). The general handwriting recognition problem presents many difficulties produced by interpersonal and intrapersonal variations when writing, the cursive nature of handwriting, the use of different pen types, or the presence of paper with noisy background [1]. Srihari et al. [2] have studied and determined with scientific rigor the individuality of handwriting. Regarding the handwriting recognition problem, there are two variants: *offline* and *online* recognition [1]. The offline problem consists in recognizing handwritten text that has previously been written on paper and then digitized. The online handwriting problem aims to recognize the text that was written using some kind of electronic digitizer device. The sensors of this device also record a set of dynamic measures about how the act of writing is produced (e.g., writing pressure, pen altitude, and azimuth). In recent years, there

has been more progress on the online modality but the offline one is still far to be solved in an unrestricted manner [3].

There exist additional complex recognition problems associated with handwriting. Automatic classification of individuals into different demographic categories [4–6] using handwriting presents interesting applications in areas such as Forensic Biometrics, Psychology, Human-Computer Interaction, or Biometric Security [7, 8]. For example, when an anonymous piece of handwritten text is found at a crime scene and it is possible to automatically recognize that the writer is a “left-handed woman,” this can reduce the group of suspects to be investigated. Psychology can also get benefits from research on handwriting style since it could be possible to identify correlations between the handwriting and some personality attributes of the writer. In the field of Human-Computer Interaction, if gender and/or handedness of a user can be automatically predicted, the computer applications could offer him/her a more personalized interaction (e.g., gender-oriented advertising). Biometric Security can also benefit from handwriting prediction since this fact can



FIGURE 1: Some samples of handwritten text lines in English from IAM database (a, c, e, and g) and in Arabic from KHATT database (b, d, f, and h). These texts were written by (a)-(b) right-handed male, (c)-(d) left-handed male, (e)-(f) right-handed female, and (g)-(h) left-handed female, respectively.

be combined with other biometric modalities in order to improve security when accessing computer systems.

These handwriting-based demographic prediction problems include gender, handedness, age ranges, or even nationality of a person [9]. This group of supervised learning problems can be considered as binary or multiclass ones. The most common binary problems are gender prediction (where handwriting texts can be classified as written by men or by women) and handedness prediction (where handwriting texts can be classified as produced by right-handed or by left-handed writers). Among the multiclass problems, one can discriminate among texts written by people included in different age intervals, in specific human races, or even in groups of nationalities. A property of all these problems is that they can be either balanced (i.e., where approximately half of the population belong to each class) as in the case of gender classification or unbalanced as in the case of the handedness classification (where the “left-handed” class only includes approximately 10% of the individuals). In general, these demographic classification problems are very complex, even for humans, since it is quite difficult to find which handwriting features properly characterize each involved class. An example of this occurs in the classification of gender. Although it is accepted that feminine writing is rounder and neater than masculine one, there are some cases where masculine writing may have a “feminine” appearance and vice versa. Figure 1 illustrates different handwriting text lines written by a “right-handed male,” a “left-handed male,” a “right-handed female,” and a “left-handed female” using two different alphabets (Latin and Arabic, resp.). In this paper, we additionally aim to analyze the relationships between the gender and handedness handwriting features.

1.1. Related Work. There are relatively few works in the literature on these problems (mostly, on the binary ones) which have been started to be investigated recently in an automatic form [9–11]. One important difficulty is that there are few handwriting databases with annotated demographic information of the writers. Other aspects that hinder this problem are similar to those presented by the general handwriting recognition problem (e.g., cursive features).

Neural networks (NN) have been applied for many years in the analysis of high-dimensional, nonlinear, and complex classification problems [12], as is the case of automatic handwriting recognition [1]. The handwriting problem has been investigated since many years using different types of NN [13, 14] for both online and offline cases [1] and even also for alphabets different from Latin (e.g., Arabic in [15]).

Two main situations can be distinguished in the automatic offline handwriting recognition of text: first, the recognition of isolated characters, which is actually solved with error rates lower than 1% [16]; second, the recognition of groups of connected characters (e.g., words or text patches), where the success rates are still far from this value. Traditionally, continuous handwriting recognition [17] from digitized documents followed a sequence of stages including preprocessing, segmentation, feature extraction, and classification [18]. Handwritten character segmentation is a particularly complex problem because it is sometimes impossible to determine where one letter ends and where the next one begins. To overcome this difficulty, holistic methods have been recently proposed, which handle each word as a whole. These solutions were usually based on hidden Markov models (HMM) [19] or neural networks (NN) [3]. In recent years, this has changed with the emergence of algorithms that

allow training deep networks presenting multiple hidden layers which are able to extract more complex and relevant features. Since each hidden layer computes a nonlinear transformation of the previous layer, a deep network can have significantly greater representational capacity (i.e., it can learn more complex functions) than a shallow network. In a 2015 survey, Patel and Thakkar [18] pointed out that a 100% success rate is still far behind in the problem of continuous handwriting recognition. Holistic methods eliminate the need to perform complex segmentation tasks on handwriting. In 2016, Bluche [20] presented a system that uses a modification of a Long Short-Term Memory (LSTM) neural network that performs the processing and recognition of complete paragraphs. However, these methods limit the vocabulary that may appear in the text. For this reason, only good recognition results are obtained in cases of limited vocabularies [18]. To break this line of reduced vocabularies, some authors are successfully employing recurring networks such as Connectionist Temporal Classification (CTC) [20, 21].

Regarding the considered demographic classification problems using handwritten texts [22, 23], gender prediction has been the most addressed one. It was studied by Graphonomics and Psychology in a nonautomatic form since the beginning of last century [24, 25]. One of the first automatic methods to classify gender from offline handwriting was presented by Hecker in 1996 [26]. Using handwriting of 96 males and 96 females and automatic pixel intensity statistics, the author achieved an overall classification rate of 71.5%. In 2003, Koppel and collaborators [27] used automatic learning algorithms with manuscript documents extracted from the British National Corpus (BNC) [28]. Each document was represented by a feature vector of characteristics, whose dimensionality was reduced by eliminating irrelevant features. Their experiments produced an average correct classification higher than 85% for gender classification. In 2004, Tomai et al. [29] applied a k -nearest neighbor (knn) classifier to microfeatures extracted from offline characters from the CEDAR letter database [2] to diverse demographic problems and reported gender classification results of around 70%. Liwicki et al. [10] proposed two online gender classification approaches, respectively, based on SVM classifier and a mixed Gaussian model (GMM). The experiments performed for the evaluation were carried out with the IAM database and showed a correct prediction of 62% with SVM and 67% with GMM in gender classification. These same authors in 2011 [30], using again GMM, obtained global accuracy results of 67.57% for both offline and online gender recognition using the IAM database. Al Maadeed and Hassaine (2014) [9] focused their research on the problem of automatic gender prediction from offline manuscripts using two approaches. In the first one, all individuals wrote the same text, while in the second one, each individual wrote a different text. From each document, they extracted a set of shape features (e.g., curvatures, chain codes, or stroke orientations) that were classified using Random Forests (RF) and Kernel Discriminant Analysis (KDA). The evaluation of the system was performed using the QUWI database [31] through different experiments with Arabic texts, English texts, and the combination of both.

Best prediction results were achieved by combining both languages and when the handwritten texts were the same, with an accuracy of 69.8% with RF and 72.3% with KDA, respectively. Bouadjenek and collaborators (2015) [11] have addressed the gender classification problem using features from Histogram of Oriented Gradients (HOG) and an SVM classifier. Their evaluation was performed using the IAM and KHATT databases, which contain handwritten documents in English and Arabic, respectively, and achieved average precision of 75.45% for IAM and 68.89% for KHATT. Siddiqi et al. (2015) published a study on gender classification from handwriting [32] which focused on features based on slant/orientation, roundedness/curvature, neatness/legibility, and writing texture. These features were classified using ANN and SVM and evaluated on the QUWI and the MSHD databases. The best classification results for the two databases were achieved using slant and curvature features with an SVM classifier (68.75% for QUWI and 73.02% for MSHD, resp.). In 2016, two studies regarding the gender classification problem were published at the ICDAR conference. A first study, by Mirza et al. [33], used texture features that were extracted using a bank of multiscale and multiorientation Gabor filters, and these features were classified with feed forward neural networks. Best experimental results reported by these authors were achieved using only Arabic texts from the QUWI dataset. A second study, by Tan and collaborators [34], proposed the extraction of multiple geometrical (e.g., local curvature of strokes) and transformed (e.g., Fourier coefficients) features and the use of Mutual Information to select an optimal subset of features in classifying the writer's gender. This study reported an average accuracy of 67.2% using ICDAR 2013 and RDF datasets. In 2017, Akbari et al. [35] proposed an effective technique to predict gender that converts a handwritten image into a textured one that is decomposed into various subbands at various levels. These subbands are used to construct Probabilistic Finite State Automata (PFSA) that generate the feature vectors. With these vectors, they trained a neural network (NN) and an SVM. To evaluate both classifiers, text-dependent and text-independent tests have been performed with the QUWI and MSHD [36] databases. Their experiments showed correct classification results of 77.8% with SVM and 79.3% with NN in the case of QUWI dataset, whereas with the MSHD dataset these results were, respectively, 79.9% with SVM and 79% with NN. Finally, also in 2017, Bouadjenek et al. [37] compared Histogram of Oriented Gradients (HOG) with Local Binary Patterns (LBP) as feature extractors for gender classification on the IAM dataset. Using separately for the extracted HOG and LBP features an SVM classifier, the HOG produced better correct gender prediction (74% versus 70%).

The problem of handedness classification from handwriting has also been more recently studied in an automatic way [24, 38]. According to Saran et al. [39], it is possible to discriminate handedness based on direction of strokes and slope of letters (i.e., left-handed writers produce strokes in right-to-left direction and the slope of letters is backwards, whereas right-handed ones produce opposite features).

Bandi and Srihari [4] in 2005 presented an online handedness system based on pen pressure and writing movement

with a classification result of 74.4%. In 2007 Liwicki et al. [10] also proposed an online method for handedness detection using SVM and GMM for classification using the IAM database and reported results of 62% with SVM and 84.6% with GMM, respectively. Al-Maadeed and others [40] studied in 2013 the offline handedness classification problem (i.e., without using dynamic information from handwriting). They extracted shape and curvature features from strokes and used a knn classifier, reporting results of 71.5% on the QUWI database (with both English and Arabic texts). A work of 2015 by Bouadjenek et al. [11] applied to handedness prediction the same offline system that they used for gender classification (i.e., HOG for feature extraction and SVM as classifier) on the KHATT dataset (also with English and Arabic texts) reporting 83,93% of success. More recently, Al-Maadeed et al. [41] have presented a novel framework for handedness detection, using offline handwriting and fuzzy logic. These authors collected a database of handwritten texts (in Arabic and English) from 121 writers and extracted a high number of shape features from the texts. A dimensionality reduction stage, based on fuzzy conceptual reduction by applying the Lukasiewicz implication, was included. The classification stage was performed using a knn method, producing an average result of 83.43% for their dataset.

Most recent works present results for more than one demographic problem using handwriting (e.g., they separately handle both gender and handedness problems; see, e.g., [10]). Other recent papers additionally include some multiclass problems like age range prediction [11, 42] and nationality [9].

1.2. Proposed Approach. In general, there is an inherent difficulty in identifying the best features to discriminate between the subclasses (e.g., men versus women) in demographic classification problems based on handwriting [29]. Some types of deep networks like convolutional neural networks can find automatically good features and also perform the classification task. Convolutional neural networks had proven better capabilities to extract relevant handwriting features when compared to using hand-crafted ones for the automatic text transcription problem.

In this paper, we describe a detailed experimental study on the application of these deep neural networks to several automatic demographic classification problems based on handwriting. In particular, we address three types of demographic problems: gender, handedness, and the combined “gender-and-handedness” classification. In order to test our proposal, two public handwriting datasets are used: IAM with English texts and KHATT containing Arabic texts.

To the best of our knowledge, our work also presents the first approach to the combined gender-and-handedness prediction, which has not been addressed before by other researchers. Moreover, this multiclass approach for gender and handedness problems produced better average accuracy results than handling successively the two binary problems. Our solution exhibits generic behavior because it has a unique configuration of convolutional neural network for the three considered demographic problems.

1.3. Contributions and Outline of the Paper. The main contributions of this work are the following ones:

- (i) This is the first paper on the application of deep networks to demographic classification problems from handwriting. A different problem is identifying a writer from his/her handwriting using deep learning models, which has been recently studied by Xing and Qiao [43]. Moreover, although there exist other deep learning approaches to predict the gender, these are based on other types of input patterns different from handwriting. For example, Bartle and Zheng [44] used stylistic information in computer blogs, and Levi and Hassner [45] used facial images.
- (ii) In addition to the separated gender and handedness classification problems from handwriting, we introduce the combined gender-and-handedness problem, where four subclasses are defined: right-handed men, left-handed men, right-handed women, and left-handed women, respectively. This novel multiclass problem, which is not handled by previous works, is more complex than separate binary gender and handedness ones, and it is of interest to Forensic Biometrics applications [8].
- (iii) For the sake of simplicity in the proposed solutions, we have designed a unique configuration of convolutional neural network, with specific parameter values for each of the three considered demographic problems.
- (iv) Our prediction method remains relatively robust for more than one considered alphabet (i.e., Latin and Arabic), and it achieved competitive classification results in two of the most used datasets for these problems: IAM and KHATT.

This paper is organized as follows. Section 2 describes the methods and materials used in this research. Section 3 describes the experimental setup, presents the results achieved for each of the considered demographic problems, and discusses these results. Finally, Section 4 summarizes the conclusions of the work.

2. Materials and Methods

In this section, we summarize some fundamentals of deep learning and convolutional neural networks. Next, the common characteristics of the proposed convolutional model, used for all considered handwritten-based demographic problems, are described. We continue with a description of the preprocessing applied to training data. Next, the specific features of the convolutional networks applied to respective gender, handedness, and combined classification problems are explained. Finally, two databases used in our experimentation are summarized.

2.1. Deep Learning and Convolutional Neural Networks. The essence of deep learning is the application to learning problems of artificial NN that contain more than two hidden layers. Deep learning has produced extraordinary advances

in difficult computational problems that have resisted the attempts of the AI community during decades. This new paradigm has been used to discover complex structures in high-dimensional data [46]. Deep learning is currently being applied to many scientific domains and, in particular to image recognition problems where it has beaten other machine-learning techniques [46].

Convolutional neural network (CNN or ConvNet) is a well-studied deep learning architecture that was inspired by the natural visual perception mechanism. LeCun and collaborators [13] presented in 1990 the framework for the CNN, and they created a multilayer network called LeNet-5 which was able to classify handwritten digits. This type of NN included three types of layers: convolutional, pooling (or subsampling), and fully connected (or dense) layers. Convolutional layers aim to learn feature representations of inputs. Each of them is composed of several convolution kernels which are used to compute different feature maps. Each neuron of a feature map is connected to a region of neighbor neurons of the previous layer. The new feature map is calculated by first convolving the input with a learned kernel and then applying an element-wise nonlinear activation function on the convolved results [47]. Note that the kernel is shared by all spatial locations of the input. The complete feature maps are obtained by using several different kernels. Each pooling layer searches to achieve shift invariance and reduces the resolution of the feature maps. It is usually placed between two convolutional layers. Finally, after several stacks of convolutional and pooling layers, there appears one or more fully connected layers which perform the final classification task. Like other multilayer networks, CNNs were trained using types of backpropagation algorithms.

However, due to the need of large training data and the lack of computing power at that time, these original LeNet-5 networks could not perform well on complex problems. In 2012, Krizhevsky et al. [48] proposed a new CNN model with a deeper structure, called ImageNet, which showed significant improvements upon other image classification methods. It included data augmentation to enlarge the training dataset, “dropout” (i.e., dropping out a percentage of neuron units, both hidden and visible) for reducing overfitting, ReLU activation function for reducing the effect of gradient vanishing during backpropagation, and the use of GPUs for accelerating the overall training process. Moreover, the application of proposed good practices [48] when designing and training convolutional networks is also important for achieving effective results.

The inputs of the CNN for our considered problems are order-3 tensors (i.e., a monochannel image with H rows and W columns). These inputs are processed sequentially through all network layers and produce as output a c -dimensional vector for the classification problem with c classes. Using some mathematical notation, the value at position (i, j) in the k -th feature map of the l -th network layer, represented as: $z_{i,j,k}^l$, can be calculated as follows:

$$z_{i,j,k}^l = w_k^l x_{i,j}^l + b_k^l \quad (1)$$

where w_k^l and b_k^l are the respective weight and bias vectors of the k -th filter in the l -th layer and $x_{i,j}^l$ is the local input region for this position and layer. Network weight masks (which define convolution kernels) w_k^l are shared, thus reducing the training time. Like other types of NN, in order to recognize nonlinear features, the value computed by (1) is passed through the ReLU activation function:

$$a_{i,j,k}^l = \text{ReLU}(z_{i,j,k}^l). \quad (2)$$

These results, produced after the inputs pass through a convolutional layer, are then processed by a pooling layer (i.e., it can be a *max-pooling* layer, placed between two convolutional layers) in order to achieve invariance and reduce the size of feature maps. New intermediate values are computed as follows:

$$y_{i,j,k}^l = \text{max_pooling}(a_{i,m,n}^l), \quad (3)$$

where $(m, n) \in \mathcal{R}_{i,j}$ represents a local neighborhood around position (i, j) . Note that kernels of lower layers can detect low-level features while kernels in higher layers detect high-level features. Finally, after several convolutional and pooling layers, there exists one or more fully connected layers, and the last one is the output layer which classifies the input test pattern into one of the predefined categories (i.e., supervised classification).

2.2. Proposed Deep Learning Architecture Framework for Demographic Problems. This subsection describes the common characteristics in our solution for the considered demographic problems using handwritten text. Next, in the successive subsections, we point out the specific aspects of each particular problem, namely, gender classification, handedness classification, and combined gender-and-handedness classification. For predicting the subclasses in the three problems, we used the same CNN architecture shown in Figure 2. The general proposed neural model has 6 trainable layers, grouped in 2 stacks of convolutional and subsampling (or max-pooling) layers, and 2 final dense layers. The network receives input images with a spatial resolution of 30×100 . After some experimentation, we used kernels of size 5×5 for the convolutional layers and of size 2×2 for the subsampling layers. These experiments showed us that smaller kernels produced worst results and bigger kernels did not improve significantly the results. Parameters a , b , and c in this figure, respectively, correspond to the number of feature maps for the first convolutional layer, the number of feature maps for the second convolutional layer, and the number of output neurons in the last layer (i.e., the problem subclasses) for each of the three demographic problems. The corresponding values of these parameters for each considered problem are detailed in Section 2.4.

In all the convolutional layers, we used zero padding to preserve the spatial size, all hidden layers include the nonlinear rectification units (ReLU), and the output layer used the *SoftMax* activation function. Dropout regularization with value of 0.25 was applied to each of the convolutional layers and with value of 0.5 to the first dense layer. The binary

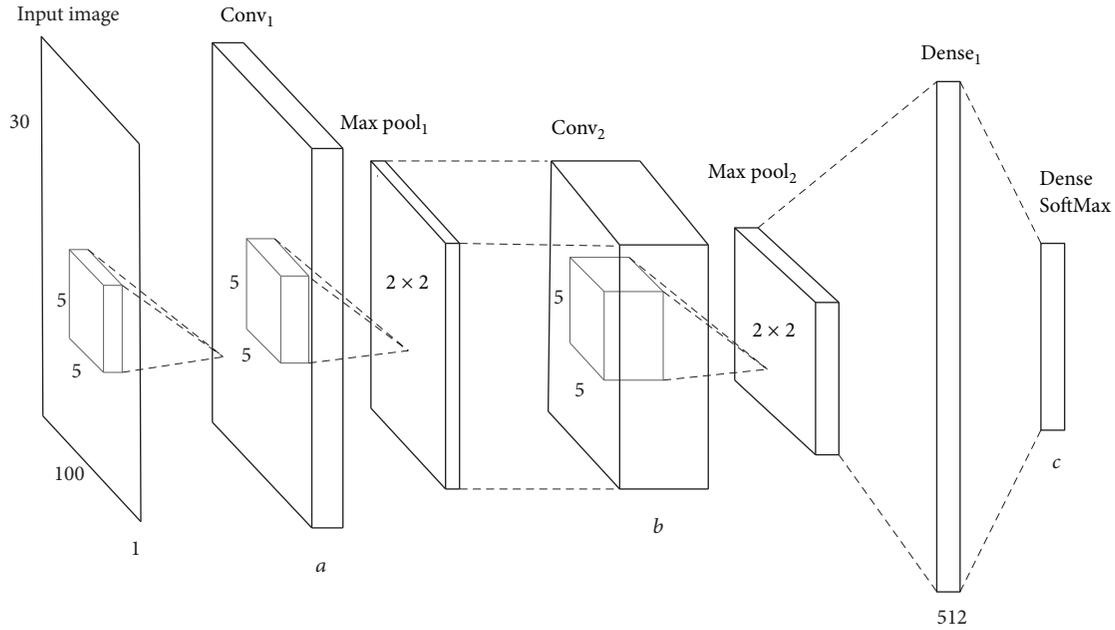


FIGURE 2: Proposed CNN architecture for demographic problems using handwriting.

models were trained using *Stochastic Gradient Descent* (SGD) and the multiclass one was trained using Adam optimization algorithms, respectively, both with a learning rate value of 0.001 and net *weight decay value* of $1E-7$. All these parameter values were determined through experimentation.

Figure 3 sketches the prediction method followed to address all demographic classification problems. Each dataset, composed by a collection of handwritten separated lines (each one with its associated demographic information), is partitioned into subsets of text images: training, validation, and test ones, respectively. There exists also separation between the “training” and “test” individuals in order to prevent the CNN model from “learning” the specific handwriting of each individual. Given one handwritten line, it is automatically splitted into their component “words” (i.e., text patches) that, after being preprocessed, will be the inputs to the network. The extraction of “words” given in a text line is computed by first applying a morphological dilation to the line, then extracting the contours from the resulting dilated binary image, and finally computing the bounding rectangles from the connected contours.

The CNN model can predict for a given unknown word its subclass in each considered problem. Finally, the predicted results of test words, contained into a text line, are combined by a majority-voting scheme to determine the final prediction result for the considered test line. The advantage of this approach is making available to the network a higher number of training samples (i.e., thus allowing it to achieve internal representations of smaller pieces of text when analyzing the involved graphisms). Moreover, we use a Learn-on-Demand method [49] when training the CNN models, thus avoid generating in advance all the possible training samples for the network.

2.3. Preprocessing of Training Data. When using deep learning neural networks in classification problems, it is necessary to have a large amount of training data (in some cases, millions) so that the network is able to discriminate correctly among the different classes. Data augmentation is an elegant solution to the problem and it consists in transforming the available data into new data without altering their nature. Some common data augmentation methods [47] are geometric transformations (such as normalization, rotation, shifting, or rescaling), morphological operations, and various photometric transformations. Of course, these transformations can be successively applied to the same input image [50].

Pseudocode 1 summarizes our data augmentation approach, which is applied to any training word image w .

Using Pseudocode 1, we produce synthetic word images as shown in Figure 4. These generated images are rescaled to be the training inputs of the CNN classifier.

2.4. Specific Model Features for Gender, Handedness and Combined Classification. Regarding our solution to the binary gender problem with a convolutional network, the used architecture configuration is presented in Figure 2 with respective parameter values of $a = 128$ (i.e., number of feature maps for the first convolutional layer), $b = 256$ (i.e., number of feature maps for the second convolutional layer), and $c = 2$ (number of output neurons or subclasses in the last layer). The number of training epochs for this problem was 200. In each epoch an amount of 100,000 synthetic training and 20,000 validation words (obtained from the original ones using the algorithm of Pseudocode 1) were presented to the network. One-half of the synthetic training and validation sets of words correspond to masculine writers and the other half to feminine ones.

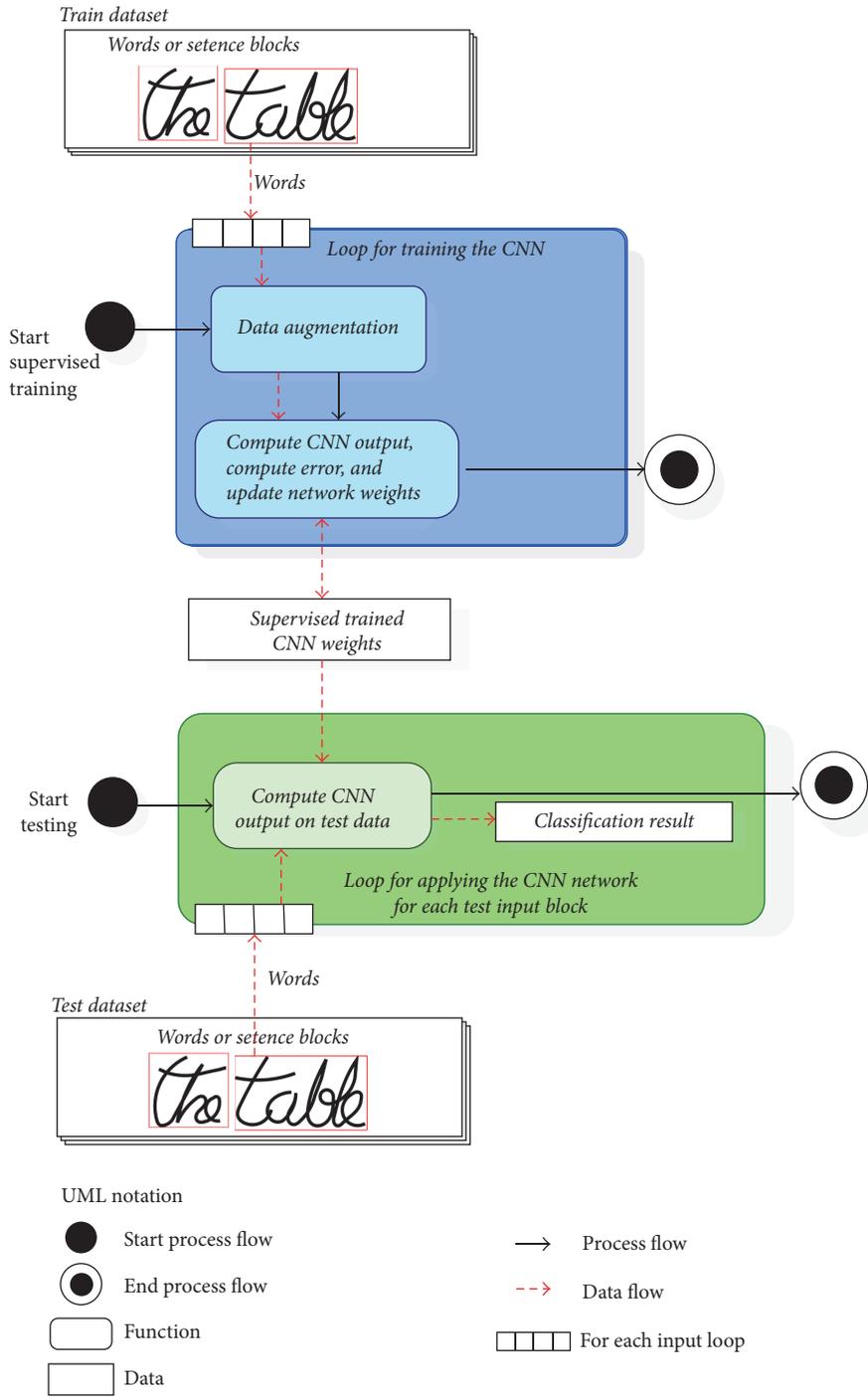


FIGURE 3: Overview of proposed method for demographic classification problems.

Handedness prediction is also a binary problem (i.e., “right-handed” and “left-handed” subclasses), where the number of original patterns in both subclasses is unbalanced for most of available datasets. In general, the databases have around 90% of samples for right-handed writers and 10% for left-handed ones, which is approximately the proportion of both subclasses in the world. The CNN architecture configuration used is the same shown in Figure 2 with respective parameter values of $a = 64$, $b = 128$, and $c = 2$. The number

of training epochs for this problem was 200. In each epoch, a total of 100,000 synthetic training and 25,000 validation words (obtained from the original ones using the algorithm of Pseudocode 1) were presented to the network. One-half of the synthetic training and validation words corresponded to right-handed writers and the other half to left-handed ones.

The combined multiclass problem categorizes the subclasses of combining gender with handedness. In particular, it needs previous partitioning of the datasets into individuals

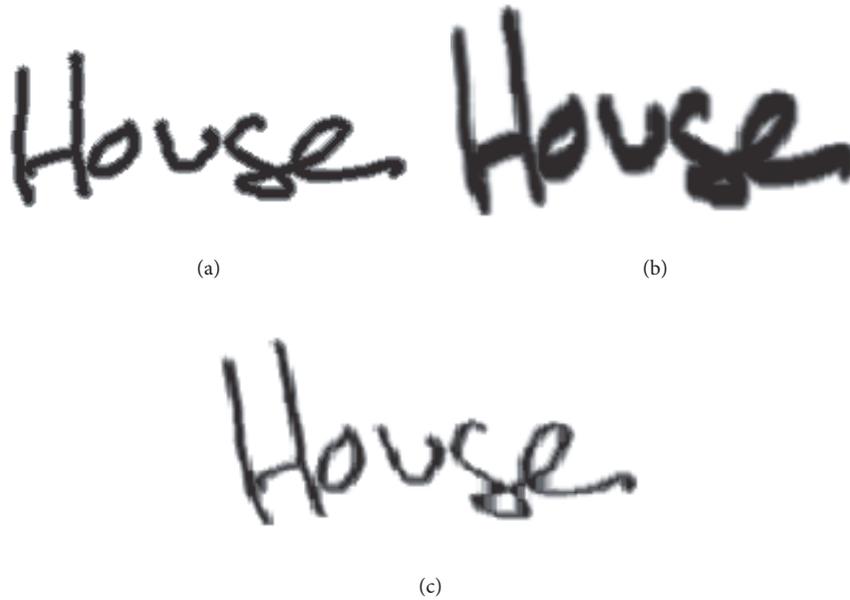


FIGURE 4: Some examples of combined transformations applied to a sample training word: (a) original image; (b) synthetic image obtained from (a) by successively applying a -5° inclination, a vertical scaling of 20%, and a morphological dilation; (c) synthetic image obtained from (a) by successively applying a -10° inclination, both vertical and horizontal scaling of 10% and a morphological erosion.

```

algorithm generate_modify_image ( $w$ ):
  # affine inclination
   $i$  = randomInclination in  $\{-10^\circ, -5^\circ, 0^\circ, 5^\circ, 10^\circ\}$ 
   $w_r$  = affine_rotation ( $w, i$ )
  # positive scaling
  vs = randomVerticalScaling in  $\{0\%, 10\%, 20\%\}$ 
  hs = randomHorizontalScaling in  $\{0\%, 10\%, 20\%\}$ 
   $w_{r,s}$  = increasing_scaling ( $w_r, vs, hs$ )
  # binary morphological filter with a  $3 \times 3$  structuring element
   $m$  = randomMorphologyOperation in  $\{\text{erosion, dilation, none}\}$ 
   $w_{r,s,m}$  = morphology_filter ( $w_{r,s}, m$ )
  # produce  $30 \times 100$  rescaled image  $w_o$  using bilinear interpolation
   $w_o$  = normalize_size ( $w_{r,s,m}$ , "bilinear") # input training image for the CNN
  return  $w_o$ 

```

PSEUDOCODE 1: Pseudocode of successive random transformations applied on training images.

who correspond to “right-handed men,” “left-handed men,” “right-handed women,” and “left-handed-women,” respectively. Regarding our convolutional network solution, we also used the CNN architecture configuration presented in Figure 2 with parameter values of $a = 32$, $b = 64$, and $c = 4$, respectively. The number of training epochs for this problem was 250. In each epoch, a total of 130,000 synthetic training and 20,000 validation words (also obtained from the original ones using the algorithm of Pseudocode 1) were presented to the network. One-quarter of synthetic training and validation words corresponded to right-handed masculine, left-handed masculine, right-handed feminine, and left-handed feminine writers, respectively.

All of our algorithms were coded in Python using the OpenCV Computer Vision library and the Keras high-level

API for neural networks. Our models were trained and tested using a NVIDIA GeForce GTX TITAN Black GPU with 6 GB of frame buffer memory.

2.5. IAM and KHATT Databases. The IAM database [51–53] was created by the Computer Vision and Artificial Intelligence Research Group in the University of Bern (Switzerland). This dataset includes both an online version and an offline one. The database is specially designed to train and test text recognizers, as well as performing identification and verification experiments for writers.

The complete version of IAM Handwriting Database 3.0 is structured as follows. A number of 657 writers contributed samples of their handwriting. There are 1,539 pages of scanned text, 5,685 isolated and labeled sentences, 13,353

TABLE 1: Number of training and text lines, per subclass and experiment, from Offline IAM database.

Gender							
Male (M)				Female (F)			
Train		Test		Train		Test	
5,228		1,659		3,234		805	
Handedness							
Right-handed (R)				Left-handed (L)			
Train		Test		Train		Test	
6,946		3,011		728		250	
Combined							
RM		LM		RF		LF	
Train		Test		Train		Test	
4,851		1,473		429		134	
2,497		1,113		297		132	

isolated and labeled text lines, and 115,320 isolated and labeled words. This dataset contains forms of unconstrained handwritten text, which were scanned at a resolution of 300 DPI and saved as PNG images with 256 gray levels. From each writer, the following information was stored in the database: the gender, native language, and other features relevant for the analysis such as if he/she is right-handed or left-handed writer.

In our experiments, we have only used a subset of the offline sentences of this dataset (which are here named as ‘‘Offline IAM’’). Table 1 shows the number of training and test lines used for each class and considered problem for the Offline IAM dataset.

The KHATT database [54, 55] was created by a research group of the King Fahd University (Saudi Arabia). It contains offline handwritten Arabic texts of approximately 1,000 writers from different countries, genders, handwriting, and educational levels. This database can be used in problems of identification of writers, techniques of binarization and elimination of noise, handwriting recognition, and techniques of line segmentation. Each of the 1,000 writers, 677 men and 323 women, wrote four paragraphs which contained a common part for all writers and a free part where each one wrote a different text. A total of 4,000 paragraphs were segmented into text lines with about 200,000 different words. In addition, 928 of the writers were right-handed and 72 were left-handed. The database also contains information related to writers such as name, age, gender, or handedness. So, it can be very useful when using the data for a particular demographic problem. Table 2 shows the number of training and test lines used for each class and problem considered for the KHATT dataset.

3. Results and Discussion

This section describes the experiments and corresponding results on the two used databases: Offline IAM and KHATT, respectively. Next, these results are compared to those presented by related works. Finally, an analysis and discussion on the achieved results are also included.

TABLE 2: Number of training and test lines used, per subclass and experiment, from KHATT database.

Gender							
Male (M)				Female (F)			
Train		Test		Train		Test	
3,931		609		1,846		358	
Handedness							
Right-handed (R)				Left-handed (L)			
Train		Test		Train		Test	
5,344		877		433		90	
Combined							
RM		LM		RF		LF	
Train		Test		Train		Test	
3,601		547		330		62	
1,743		330		103		28	

In order to evaluate our approach, we use some standard performance metrics for binary and multiclass categorization. These measures, which are calculated for each subclass of a given demographic problem, are precision, recall, and F -measure. They are defined for a binary problem and given subclass c as given by

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c} \quad (4)$$

$$\text{Recall}_c = \frac{TP_c}{TP_c + FN_c} \quad (5)$$

$$F\text{-measure}_c = 2 \frac{\text{Precision}_c \times \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c}, \quad (6)$$

where TP_c , FP_c , and FN_c are, respectively, the number of true positives, false positives, and false negatives in the class c .

The overall accuracy of the binary model can be directly computed from any of the two classes c , since it has the same value for the two classes due to the exchange of positives and negatives property [56]. This accuracy value is computed as follows:

$$\text{Overall}_{\text{Accuracy}} = \frac{TP_c + TN_c}{TP_c + TN_c + FP_c + FN_c}. \quad (7)$$

The previous formulae can be extended to multiclass categorization problems [56]. Definitions of Precision_c and Recall_c are now adapted for our 4-class combined demographic problem. Given the confusion matrix $M = [(M_{i,j}) \mid 1 \leq i, j \leq 4]$ corresponding to our multiclass problem, these metrics are now computed as follows:

$$\text{Precision}_c = \frac{M_{c,c}}{\sum_j M_{j,c}} \quad (8)$$

$$\text{Recall}_c = \frac{M_{c,c}}{\sum_j M_{c,j}}$$

The expression of F -measure for each class in the multiclass problem is computed using (6) but with accuracy and precision values, respectively, computed by (8). Finally,

TABLE 3: Performance results for gender classification on IAM database.

Male (M)	Precision	86.81
	Recall	84.14
	<i>F</i> -measure	85.46
Female (F)	Precision	69.26
	Recall	73.66
	<i>F</i> -measure	71.39
Overall accuracy		80.72

TABLE 4: Performance results for handedness classification on IAM database.

Right-handed (R)	Precision	96.75
	Recall	93.02
	<i>F</i> -measure	94.85
Left-handed (L)	Precision	42.60
	Recall	63.40
	<i>F</i> -measure	50.64
Overall accuracy		90.70

the average accuracy for the multiclass problem [56] can be computed as follows:

Average Accuracy

$$= \frac{\sum_{c=1}^l ((TP_c + TN_c) / (TP_c + TN_c + FP_c + FN_c))}{l}, \quad (9)$$

where the respective accuracy values of all l classes (with $l = 4$) are averaged.

In our context, the precision of a subclass c is the quotient between the number of correctly classified handwritten text lines into the subclass c and the total number of text lines classified into this subclass c . The recall of a subclass c is the quotient between the number of correctly classified handwritten text lines into the subclass c and the number of text lines that truly belong to the class c . The *F*-measure combines precision and recall and reflects the relative importance of recall with respect to the precision. Finally, the average accuracy represents a global measure of the classifier’s performance for each considered problem. As recommended by [56] for binary and multiclass classification, the previous evaluation measures have been applied to determine the performance of our proposals.

3.1. Experiments Using the Offline IAM and KHATT Datasets.

The previous evaluation measures have been applied to determine the performance of our models in the considered demographic prediction problems using English and Arabic texts. Tables 3, 4, and 5, respectively, present the calculated scores (in %) for the gender, handedness, and combined problem using the Offline IAM dataset, according to the measures given by (4)–(9).

Note that if both binary gender and handedness problems were independently handled, the joint average accuracy produced by the corresponding classification models should

TABLE 5: Performance results for combined gender-and-handedness classification on IAM database.

Right-handed male (RM)	Precision	69.98
	Recall	76.10
	<i>F</i> -measure	72.91
Left-handed male (LM)	Precision	53.38
	Recall	52.99
	<i>F</i> -measure	53.18
Right-handed female (RF)	Precision	62.51
	Recall	58.58
	<i>F</i> -measure	60.48
Left-handed female (LF)	Precision	66.22
	Recall	37.12
	<i>F</i> -measure	47.57
Average accuracy		83.19

TABLE 6: Performance results for gender classification on KHATT database.

Male (M)	Precision	71.21
	Recall	85.14
	<i>F</i> -measure	77.55
Female (F)	Precision	62.11
	Recall	41.43
	<i>F</i> -measure	49.70
Overall accuracy		68.90

be the product of their individual accuracies. This would produce, using overall accuracy values given in Tables 3 and 4, an average accuracy of 73.21%. This result is worse than 83.19% (see Table 5) obtained when we train a unique 4-class combined classification system. This fact, together with the economy in training times, shows that the proposed combined multiclass approach for the two considered problems is more effective than independently solving one binary problem and on the first classification apply the second one (i.e., in a hierarchical fashion).

Due to a substantially lower number of original training images in the KHATT database, we have applied the *transfer learning* technique (also known as *inductive training* or *pretraining*) in order to improve the classifications results for such dataset. This pretraining was only applied to the handedness and combined gender-and-handedness problems. Instead of randomly initializing the weights of the CNN connections, we have used the pretrained models built for the Offline IAM database and, after that, trained with them the respective networks with the corresponding training patterns of the KHATT dataset. This way, the knowledge gained for the CNN, while learning to recognize handwritten words with IAM, is transferred to the KHATT network. This practice is common for these networks (e.g., ImageNet) because many datasets do not have the sufficient size for enabling convolutional networks to extract relevant features for producing good classification results.

Tables 6, 7, and 8, respectively, present the calculated scores (in %) for the gender, handedness, and combined

TABLE 7: Performance results for handedness classification on KHATT database.

Right-handed (R)	Precision	92.18
	Recall	74.24
	<i>F</i> -measure	82.24
Left-handed (L)	Precision	13.34
	Recall	38.64
	<i>F</i> -measure	19.83
Overall accuracy		70.91

TABLE 8: Performance results for combined gender-and-handedness classification on KHATT database.

Right-handed male (RM)	Precision	69.23
	Recall	44.07
	<i>F</i> -measure	53.86
Left-handed male (LM)	Precision	11.84
	Recall	14.75
	<i>F</i> -measure	13.14
Right-handed female (RF)	Precision	41.81
	Recall	44.27
	<i>F</i> -measure	43.01
Left-handed female (LF)	Precision	3.76
	Recall	25.93
	<i>F</i> -measure	6.57
Average accuracy		70.84

problem using the KHATT dataset (i.e., Arabic script), according to the measures given by (4)–(9).

Note that the same fact with respect to proposed combined classification happens for the KHATT database. If both binary gender and handedness problems were independently handled, the average accuracy (obtained from values of Tables 6 and 7) is 48.86%. This result is much worse than 70.84% obtained when we train a unique 4-class combined classification system. By averaging the improvement accuracy rates of both Offline IAM and KHATT databases, our multiclass approach improved the accuracy by 29.26% when compared to separately and successively handling both binary problems. Perhaps, when training convolutional networks independently for the two demographic problems, the networks are not able to discover handwriting features which capture the interconnections between both individual problems. Moreover, when training a network for the combined multiclass problem, these related handwritten characteristics are better discovered.

With respect to the training times required for the convolutional network models using the IAM dataset, the gender problem used 100,000 training sample images and other 25,000 ones for validation (of the training). These steps were performed during 200 epochs (i.e., about 61 hours). Similar training times were required for the other two considered problems using the IAM dataset. When training our models for the KHATT dataset, the training times were significantly increased since, as explained, we first applied a pretraining

of the convolutional networks with the images of the IAM dataset.

3.2. Comparison with Related Works. Comparing our research results to those ones published on the same gender and handedness problems using the same datasets is difficult because of the differences in experimental aspects and the way the classification results are reported. Differences in experimental aspects are as follows: the different number and distribution of original images between the categories for training, validating, and testing the classification systems; the different alphabets used; the usage of the same texts written by all the writers or of different texts for each writer; and/or whether there is any preprocessing on the original datasets images. With respect to published results, there are several works [9, 35] that only report an overall accuracy result for each classification method used in the problems. In the common case of unbalanced classes, as is the case of “left-handed” in the handedness problem, this overall accuracy is not appropriate and specific measures per each class are more convenient.

Taking into account the previous remarks, we have compared our results with those reported in [10, 11, 30] which use the same databases and the same performance measures per class. The analyzed results are presented in Table 9, and they show that our approach produces the best scores in the gender problem for both IAM and KHATT databases, while the results presented in [11] are the best ones for the handedness problems for the considered datasets.

3.3. Analysis and Discussion. The analysis of our experimental results for the three considered handwritten-based demographic problems on the IAM and KHATT datasets have raised the following aspects:

- (i) The proposed combined multiclass approach for gender and handedness problems produced better average accuracy results than handling successively the two binary problems.
- (ii) Our common convolutional architecture framework for the three demographic problems has produced acceptable prediction results, even for the combined gender-and-handedness prediction problem, where there are fewer training text lines in the involved subclasses.
- (iii) Classification results on the KHATT database are worse than the corresponding ones on the Offline IAM database. This can be caused by the more reduced number of original training examples in the Arabic dataset. In spite of applying data augmentation and transfer learning as optimization techniques for improving the classification results of the convolutional networks, we noticed that, when there is a more reduced number of original training samples (i.e., those ones provided by the dataset without data augmentation), the prediction results are worse.
- (iv) We have to remark the importance of transfer learning (or pretraining) when training convolutional networks in problems with a reduced number of original

TABLE 9: Comparison of overall accuracy results for gender and handedness problems on Offline IAM and KHATT databases presented in related papers.

	Liwicki et al. [30]		Liwicki et al. [10]		Bouadjenek et al. [11]		Our approach	
	Gender	Handedness	Gender	Handedness	Gender	Handedness	Gender	Handedness
IAM	67.57	84.66	75.45	100.0	80.72	90.70		
KHATT			68.89	83.93	68.90	70.91		

samples per class. This is the case of “left-handed” in the KHATT database.

- (v) Some papers addressing demographic problems from handwriting only report global accuracy results in classification. However, these results are not valuable at all when the number of patterns per subclass is highly unbalanced (e.g., “left-handed”). It is more important to report the correct prediction results per class (i.e., using precision and recall measures).

Next, discussion of results is completed in several directions: complexity of the proposed model respect classical approaches, necessity of data augmentation, and computing times, respectively. Regarding the complexity of the proposed model with respect to classical approaches (i.e., feature-based ones), from a developer’s viewpoint using convolutional neural networks (CNN) is simpler than determining which features are the best ones for discriminating each class. Differently from other analyzed feature-based proposals (see, e.g., [11, 35, 40]), when using CNN one does not have to discover which features are relevant to solve the problem (i.e., this approach is a drop-in replacement to hand-made feature descriptors). Since these good internal representations are now found by the network, the model is much simpler and powerful at the same time. Regarding data augmentation, it is true that these networks require a very high number of training examples to learn well the involved classes. These examples were obtained synthetically, by creating word images through applying combinations of multiple transformations with different parameters on the original training images. In our approach, the considered transformations were random left/right sloping, vertical/horizontal scaling, and morphological erosion/dilation. With respect to training times, despite advances in CNNs these models are still highly time consuming. However, and as is a common practice, we have drastically reduced the training times of our neural networks using a cluster of GPUs.

4. Conclusion

This paper presented a detailed experimental study on the application of deep neural networks to several automatic demographic classification problems based on handwriting. In particular, we have addressed three problems: gender, handedness, and the combined “gender-and-handedness” classification. We tested our proposal on two public handwriting datasets (IAM with English texts and KHATT containing Arabic texts). Convolutional neural networks had proven better capabilities to extract relevant handwriting

features when compared to using hand-crafted ones for the automatic text transcription problem. Our work also tackled the combined gender-and-handedness prediction, which has not been addressed before by other researchers. Moreover, this combined multiclass approach for gender and handedness problems produced better average accuracy results than handling successively the two binary problems. Our solution exhibits generic behavior because it has a unique configuration of convolutional neural network for the three considered demographic problems. Finally, the comparison of these results to other connected works reveals that our solution produced the best accuracy results for the gender classification problem on both tested handwriting databases.

In summary, the advantages and novel aspects of our proposal are the following ones:

- (1) To the best of our knowledge, this is the first paper on the application of deep networks to demographic classification problems from handwriting.
- (2) We introduce and effectively address the combined multiclass “gender-and-handedness” problem.
- (3) Our approach only used a unique configuration of convolutional neural network, with specific parameter values for the three considered demographic problems.
- (4) Finally, the proposed gender/handedness prediction method remains relatively robust for more than one alphabet, and it achieves competitive classification results for two of the most used datasets in this problem: IAM and KHATT.

Future work will include the extension of this research to additional handwriting datasets containing texts written in other alphabets. We are also interested in studying new additional multiclass handwritten-based problems, especially the age prediction one. Another planned future research is the adaptation of our proposed framework in order to predict some types of demographic information from writers, which is present in historical handwritten documents.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research has been supported by the Spanish Ministerio de Economía y Competitividad (MINECO), under the Projects TIN2014-57458-R and TIN2017-85221-R.

References

- [1] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000.
- [2] S. N. Srihari, S.-H. Cha, H. Arora, and S. Lee, "Individuality of Handwriting," *Journal of Forensic Science*, vol. 47, no. 4, pp. 856–872, 2002.
- [3] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Proceedings of the 21st Intl. Conf. Neural Information Processing Systems (NIPS'08)*, pp. 545–552, 2008.
- [4] K. R. Bandi and S. N. Srihari, "Writer demographic classification using bagging and boosting," in *Proceedings of the Intl. Graphonomics Society Conference (IGS'05)*, pp. 133–137, 2005.
- [5] S.-H. Cha and S. N. Srihari, "A priori algorithm for sub-category classification analysis of handwriting," in *Proceedings of the 6th International Conference on Document Analysis and Recognition, ICDAR 2001*, pp. 1022–1025, IEEE, Seattle, WA, USA, September 2001.
- [6] S. Hamid and K. M. Loewenthal, "Inferring gender from handwriting in Urdu and English," *The Journal of Social Psychology*, vol. 136, no. 6, pp. 778–782, 1996.
- [7] J. Chapran, "Biometric writer identification: Feature analysis and classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 20, no. 4, pp. 483–503, 2006.
- [8] L. Ballard, D. Lopresti, and F. Monrose, "Evaluating the security of handwriting biometrics," in *Proceedings of the Proc.*, pp. 461–466, 2006.
- [9] S. Al Maadeed and A. Hassaine, "Automatic prediction of age, gender, and nationality in offline handwriting," *Eurasip Journal on Image and Video Processing*, vol. 2014, article no. 10, 2014.
- [10] M. Liwicki, A. Schlappach, P. Loretan, and H. Bunke, "Automatic detection of gender and handedness from on-line Handwriting," in *Proceedings of the 13th Conf. Intl Graphonomics Society*, 2007.
- [11] N. Bouadjenek, H. Nemmour, and Y. Chibani, "Histogram of Oriented Gradients for writer's gender, handedness and age prediction," in *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications, INISTA 2015*, IEEE, Madrid, Spain, August 2015.
- [12] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 30, no. 4, pp. 451–462, 2000.
- [13] Y. LeCun et al., "Handwritten digit recognition with a back-propagation network in," in *Advances in neural information processing systems 2*, pp. 396–404, Morgan Kaufmann Publishers, 1990.
- [14] B. Gosselin, "Multilayer perceptions combination applied to handwritten character recognition," *Neural Processing Letters*, vol. 3, no. 1, pp. 3–10, 1996.
- [15] M. T. Parvez and S. A. Mahmoud, "Offline arabic handwritten text recognition: A survey," *ACM Computing Surveys*, vol. 45, no. 2, article no. 23, 2013.
- [16] U. Meier, D. C. Cireşan, L. M. Gambardella, and J. Schmidhuber, "Better digit recognition with a committee of simple neural nets," in *Proceedings of the 11th International Conference on Document Analysis and Recognition, ICDAR*, pp. 1250–1254, September 2011.
- [17] A. L. Koerich, R. Sabourin, and C. Y. Suen, "Large vocabulary off-line handwriting recognition: a survey," *Pattern Analysis & Applications*, vol. 6, no. 2, pp. 97–121, 2003.
- [18] M. Patel and S. P. Thakkar, "Handwritten character recognition in english: a survey," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 2, pp. 345–350, 2015.
- [19] T. Plötz and G. A. Fink, *Markov Models for Handwriting Recognition*, Springer Briefs in Computer Science, Springer, London, UK, 2011.
- [20] T. Bluche, "Joint Line Segmentation and Transcription for End-to-End Handwritten Paragraph Recognition," in *Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS'16)*, Allen Institute for Artificial Intelligence, Barcelona, Spain, 2016.
- [21] O. Nina, *Connectionist Temporal Classification for Offline Handwritten Text Recognition*, BYU Conference Center, 2016.
- [22] R. J. Klimoski and A. Rafaeli, "Inferring personal qualities through handwriting analysis," *Journal of Occupational Psychology*, vol. 56, no. 3, pp. 191–202, 1983.
- [23] T. Dziejczak, "Right hand writing vs. Left hand writing of one person. A comparative study," *Z Zagadnien Nauk Sadowych*, vol. 94, pp. 564–577, 2013.
- [24] R. A. Huber and A. M. Hendrick, *Handwriting Identification: Facts and Fundamentals*, CRC Press, Boca Raton, FL, USA, 1999.
- [25] E. S. Nogueras, M. F. Zanuy, and J. R. Alcobé, "Gender classification by means of online uppercase handwriting: a text-dependent allographic approach," *Cognitive Computation*, vol. 8, no. 1, pp. 15–29, 2016.
- [26] R. Hecker, "The scientific examination of sex differences," Proc. Fifty-Fourth Annual Meeting of the American Society of Questioned Document Examiners, 1996.
- [27] M. Koppel, S. Argamon, and A. R. Shimoni, "Automatically categorizing written texts by author gender," *Literary and Linguistic Computing*, vol. 17, no. 4, pp. 401–412, 2003.
- [28] A. G. Burnage and G. Baguley, The British National Corpus, Library and Information Briefings, 1996, <http://www.natcorp.ox.ac.uk/archive/papers/gblibs.html>.
- [29] C. I. Tomai, D. Kshirsagar, and S. N. Srihari, "Group Discriminatory power of handwritten characters," *Document Recognition and Retrieval XI, Proc. of SPIE-IS & T Electronic Imaging*, vol. 5296, pp. 116–123, 2004.
- [30] M. Liwicki, A. Schlappach, and H. Bunke, "Automatic gender detection using on-line and off-line information," *PAA. Pattern Analysis and Applications*, vol. 14, no. 1, pp. 87–92, 2011.
- [31] S. Al Maadeed, W. Ayouby, A. Hassaine, and J. M. Aljaam, "QUWI: An Arabic and English handwriting dataset for offline writer identification," in *Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition, ICFHR 2012*, pp. 746–751, Italy, September 2012.
- [32] I. Siddiqi, C. Djeddi, A. Raza, and L. Souici-meslati, "Automatic analysis of handwriting for gender classification," *Pattern Analysis and Applications*, vol. 18, no. 4, pp. 887–899, 2015.
- [33] A. Mirza, M. Moetesum, I. Siddiqi, and C. Djeddi, "Gender classification from offline handwriting images using textural features," in *Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016*, pp. 395–398, IEEE, Shenzhen, China, October 2016.
- [34] J. Tan, N. Bi, C. Y. Suen, and N. Nobile, "Multi-feature selection of handwriting for gender identification using mutual information," in *Proceedings of the 15th International Conference on*

- Frontiers in Handwriting Recognition, ICFHR 2016*, pp. 578–583, IEEE, Shenzhen, China, October 2016.
- [35] Y. Akbari, K. Nouri, J. Sadri, C. Djeddi, and I. Siddiqi, “Wavelet-based gender detection on off-line handwritten documents using probabilistic finite state automata,” *Image and Vision Computing*, vol. 59, pp. 17–30, 2017.
- [36] C. Djeddi, A. Gattal, L. Souici-Meslati, I. Siddiqi, Y. Chibani, and H. El Abed, “LAMIS-MSHD: a multi-script offline handwriting database,” in *Proceedings of the 14th International Conference on Frontiers in Handwriting Recognition, ICFHR 2014*, pp. 93–97, IEEE, Heraklion, Greece, September 2014.
- [37] N. Bouadjenek, H. Nemmour, and Y. Chibani, “Writer’s gender classification using HOG and LBP features,” *Lecture Notes in Electrical Engineering*, vol. 411, pp. 317–325, 2017.
- [38] R. N. Morris, *Forensic Handwriting Identification: Fundamental Concepts and Principles*, Academic Press, Cambridge, Massachusetts, USA, 2000.
- [39] V. Saran, S. Kumar, A. K. Gupta, and S. Ahmad, “Differentiation of handedness of writer based on their strokes and characteristic features,” *Journal of Forensic Research*, vol. 4, no. 5, Article ID 1000204, pp. 1–3, 2013.
- [40] S. Al-Maadeed, F. Ferjani, S. Elloumi, A. Hassaine, and A. Jaoua, “Automatic handedness detection from off-line handwriting,” in *Proceedings of the 2013 7th IEEE GCC Conference and Exhibition, GCC 2013*, pp. 119–124, IEEE, Doha, Qatar, November 2013.
- [41] S. Al-Maadeed, F. Ferjani, S. Elloumi, and A. Jaoua, “A novel approach for handedness detection from off-line handwriting using fuzzy conceptual reduction,” *Eurasip Journal on Image and Video Processing*, vol. 2016, no. 1, article no. 1, pp. 1–14, 2016.
- [42] N. Bouadjenek, H. Nemmour, and Y. Chibani, “Age, gender and handedness prediction from handwriting using gradient features,” in *Proceedings of the 13th International Conference on Document Analysis and Recognition, ICDAR 2015*, pp. 1116–1120, IEEE, Tunis, Tunisia, August 2015.
- [43] L. Xing and Y. Qiao, “DeepWriter: A multi-stream deep CNN for text-independent writer identification,” in *Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition, ICFHR 2016*, pp. 584–589, China, October 2016.
- [44] A. Bartle and J. Zheng, “Gender classification with deep learning,” Stanford cs224d Course Project Report, 2015.
- [45] G. Levi and T. Hassner, “Age and gender classification using convolutional neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW 2015*, pp. 34–42, IEEE, Boston, MA, USA, June 2015.
- [46] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [47] S. Albelwi and A. Mahmood, “A framework for designing the architectures of deep convolutional neural networks,” *Entropy*, vol. 19, pp. 1–20, 2017.
- [48] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Proc. Advances in Neural Information Processing Systems (NIPS’12)*, vol. 25, pp. 1097–1105, 2012.
- [49] F. Khosrowshahi, “Innovation in artificial neural network learning: Learn-On-Demand methodology,” *Automation in Construction*, vol. 20, no. 8, pp. 1204–1210, 2011.
- [50] S. Zhang, C. Zhang, and Q. Yang, “Data preparation for data mining,” *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 375–381, 2003.
- [51] U.-V. Marti and H. Bunke, “The IAM-database: an english sentence database for offline handwriting recognition,” *International Journal on Document Analysis and Recognition*, vol. 5, no. 1, pp. 39–46, 2003.
- [52] M. Liwicki and H. Bunke, “IAM-OnDB - An on-line English sentence database acquired from handwritten text on a whiteboard,” in *Proceedings of the 8th International Conference on Document Analysis and Recognition*, pp. 956–961, IEEE, Seoul, South Korea, September 2005.
- [53] IAM Homepage, <http://www.fki.inf.unibe.ch/databases/iam-on-line-handwriting-database>.
- [54] S. A. Mahmoud, I. Ahmad, M. Alshayeb et al., “KHATT: Arabic offline Handwritten Text Database,” in *Proceedings of the 13th International Conference on Frontiers in Handwriting Recognition, ICFHR 2012*, pp. 449–454, IEEE, Bari, Italy, September 2012.
- [55] KHATT Homepage, <http://khatt.ideas2serve.net>.
- [56] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

Research Article

Dynamic Traffic Congestion Simulation and Dissipation Control Based on Traffic Flow Theory Model and Neural Network Data Calibration Algorithm

Li Wang,¹ Shimin Lin,² Jingfeng Yang,^{3,4} Nanfeng Zhang,⁵ Ji Yang,³ Yong Li,³ Handong Zhou,⁴ Feng Yang,^{2,3} and Zhifu Li⁶

¹School of Computer Software, Tianjin University, Tianjin 300072, China

²Department of Computer Science and Engineering, Faculty of Engineering, The Chinese University of Hong Kong, Shatin 999077, Hong Kong

³Open Laboratory of Geo-Spatial Information Technology and Application of Guangdong Province, Guangzhou Institute of Geography, Guangzhou 510070, China

⁴Guangzhou Yuntu Information Technology Co., Ltd., Guangzhou 510665, China

⁵Guangzhou Entry-Exit Inspection and Quarantine Bureau, Guangzhou 510623, China

⁶School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou 510006, China

Correspondence should be addressed to Jingfeng Yang; jingfengyang@126.com and Zhifu Li; sundylzf@gmail.com

Received 30 June 2017; Accepted 16 October 2017; Published 31 December 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Li Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Traffic congestion is a common problem in many countries, especially in big cities. At present, China's urban road traffic accidents occur frequently, the occurrence frequency is high, the accident causes traffic congestion, and accidents cause traffic congestion and vice versa. The occurrence of traffic accidents usually leads to the reduction of road traffic capacity and the formation of traffic bottlenecks, causing the traffic congestion. In this paper, the formation and propagation of traffic congestion are simulated by using the improved medium traffic model, and the control strategy of congestion dissipation is studied. From the point of view of quantitative traffic congestion, the paper provides the fact that the simulation platform of urban traffic integration is constructed, and a feasible data analysis, learning, and parameter calibration method based on RBF neural network is proposed, which is used to determine the corresponding decision support system. The simulation results prove that the control strategy proposed in this paper is effective and feasible. According to the temporal and spatial evolution of the paper, we can see that the network has been improved on the whole.

1. Introduction

Traffic congestion is a common problem in many countries, especially in big cities. As far as our country is concerned, it is gradually spread to the medium cities and even small cities. This has brought a huge negative impact on the economic and social development of the region. For travelers, traffic congestion will lead to increased travel time, travel costs, and the quality of travel. From the point of view of traffic management, traffic congestion will result in higher operating costs and low efficiency of traffic network. At the social level, traffic congestion will cause traffic accidents, air pollution,

noise pollution, and other environmental issues. Research on the formation and propagation of traffic congestion is a new technology to control the traffic jam. The sources of traffic congestion can be summarized into three categories: (1) the temporary barrier; (2) the bottleneck of the network's capacity which is the bottleneck; (3) the random fluctuation of a particular region in the network. In this paper, traffic congestion under accident conditions is the first kind of traffic congestion caused by traffic congestion. All known and unknown events affecting the road traffic flow are collectively referred to as traffic accidents [1]. At present, China's urban road traffic accidents occur frequently, the

occurrence frequency is high, the accident causes traffic congestion, and accidents cause traffic congestion and vice versa. The occurrence of traffic accidents usually leads to the reduction of road traffic capacity and the formation of traffic bottlenecks, causing the traffic congestion. If the accident cannot be cleared in time, the vehicle will be up to the upper section of the line after the queue. If there is no timely evacuation, traffic congestion will quickly spread on the network, so as to increase the delay time of the traveler. Studies have shown that exiting road canalization is unreasonable, it is easy to form closed-loop congestion phenomenon in the dissipation of congestion. Congestion closed loop is the formation of the queue of the tail back to the congestion of the starting point. Once a closed loop is formed, traffic congestion itself is difficult to dissipate in the absence of artificial intervention [2].

In this paper, the formation and propagation of traffic congestion are simulated by using the improved medium traffic model, and the control strategy of congestion dissipation is studied. As the traffic congestion is mainly caused by accidents, temporary traffic control measures are the prohibition of the left, the prohibition of the right, and the right to ban. Vehicle ban measures can make originally lined up waiting vehicle restarted and select the way leaving the congested area. More important, timely application of vehicle turn prohibition measures can effectively avoid the formation of closed-loop congestion; even under conditions of closed-loop congestion vehicles are prohibited from turning to relieve the closed-loop congestion, so as to avoid congestion of malignant spread [3].

From the point of view of quantitative traffic congestion, the paper provides the fact that the simulation platform of urban traffic integration is constructed and a feasible data analysis method is proposed, which is used to determine the corresponding decision support system. The model is based on the international application of extensive and sophisticated series of software, which is based on learning and parameter calibration [4–11], optimization of macro simulation model, maintenance, macro (network), and medium (node and intersection). Through the platform and model, it forms the traffic state analysis of data support and auxiliary analysis platform, improving the urban road traffic forecasting ability, to provide the necessary technical support for the establishment of the urban project construction traffic impact assessment system.

2. Traffic Flow Theory Model and Data Calibration Algorithm

2.1. Traffic Flow Theory Introduction. Different traffic models can play a role in different aspects. From space, the macro traffic model can be used to analyze the main road network in the whole area, and the medium traffic model can be used to analyze a certain local project area, which can be used to support traffic planning and traffic engineering. In time, the macro traffic model takes a day or a peak hour as the scale; the medium traffic model is built on a smaller scale (e.g., every hour, every 15 minutes, and every minute); and the

microscopic traffic simulation model can be fine to the second unit (e.g., every second, every 0.1 seconds). The medium traffic model is based on the four steps of traffic planning in the macroscopic traffic model. By using the information of the basic road network, the traffic demand information is updated and the traffic assignment method is used to achieve the traffic model to express the actual state of the road network. Medium traffic model is more detailed, in addition to primary and secondary roads and main roads, and it also includes a small road. In the medium traffic model, it is necessary to take account of its drainage and traffic control methods and to consider the signal timing scheme for signal control intersection. Medium traffic model is the second level of traffic research as a supplementary tool, direct simulation of the macro background, the development and adjustment of the land, the layout of the land, and the improvement of the regional transportation system.

The micro model reflects and simulates the operation process of a single vehicle in the road network, directly simulating the operation of traffic guidance and signal control. The microscopic traffic simulation model is further refined to a number of intersections, and the parameters of each lane are set up, which can be analyzed in detail. In time, the microscopic traffic simulation model can be refined to a second as a unit (e.g., every 0.1 seconds). As the auxiliary tool of the third levels of traffic research, the model of the onlookers is analyzed and compared with the indexes of the vehicle running process, and the detailed design scheme of the local traffic improvement measures and signal optimization measures are evaluated and analyzed [7]. From the perspective of traffic demand and traffic supply, the medium traffic model is to analyze the causes of traffic congestion and to analyze the congestion control scheme.

2.2. Meso Traffic Dynamics Model. In this study, the Zhuzilin area is used to establish an example of a mesoscopic traffic model for congestion cause analysis and to make recommendations for improvement, comparison, and analysis of schemes. As this project is mainly integrated model of the technical analysis method research, the results may be different from the actual traffic conditions.

The traffic flow theory model can be divided into macro and micro methods. In view of the dynamic model, the dynamic model can meet the requirements of accuracy and has good real-time performance, so it is a hot spot in the field of intelligent transportation.

2.2.1. Queuing Theory. According to the queuing theory, the queuing delay of the queue in the i car is

$$\frac{i}{c}, \quad (1)$$

which c represents lane group output capacity (such as vehicle queue dissipation rate; traffic control signal influence on traffic flow can reflect on the change of C); ct is number of vehicles for departure time t ; if a mobile vehicle at time t reached the tail of the queue, its position is

$$q(t) = q(0) + l(ct - m), \quad (2)$$

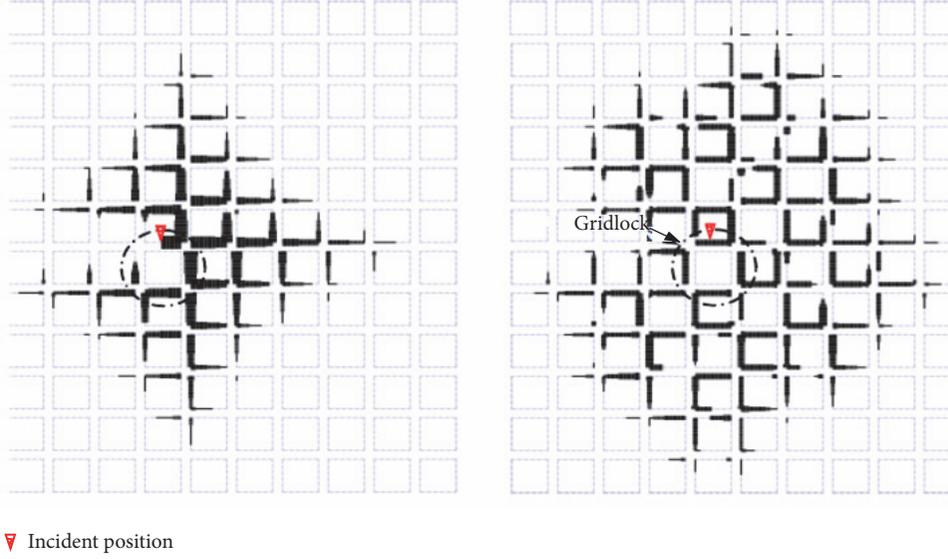


FIGURE 1: Traffic congestion.

where $q(0)$ is the location; the end of the time $t = 0$, and l from 1 jam is the average length of the vehicle. $l = 1/\rho_{\text{jam}}$ Jam is crowded density; m is the number of vehicles in time $t = 0$ between the inspection of vehicles and the end of the queue of vehicles running. In fact, the number of vehicles entering the queue before the vehicle is considered.

It is worth noting that the applicable conditions of the model are

$$0 < q(t) < L. \quad (3)$$

2.2.2. Speed Model. In the medium traffic simulator, the traffic network is divided into the connecting line, the node, and the loading point. Each connecting line is divided into a plurality of segments, each segment containing a moving element and a queue element [12, 13]. The speed of the vehicle in the moving element is determined by the velocity density model. The model is based on the assumption that the upstream part of the section is kept constant, the downstream part contains a deceleration area, and the speed of the vehicle in the region varies linearly with the position (as shown in Figure 1), where v_u is the speed of the end of the section of the section, v_d is the speed of the end of the section of the section, and L_s is the length of the deceleration area. L_s is related to the terrain features and traffic conditions of the section.

The upstream end of the road is set at 0 points, and the length of the section is L , then the downstream end of the road is located at the position of L , which establishes the relationship between the speed and position

$$v(z) = \begin{cases} v_u, & 0 \leq z \leq L - L_s, \\ \lambda(z - L) + v_d, & L - L_s \leq z, \end{cases} \quad (4)$$

$$\lambda = \frac{v_d - v_u}{L_s}.$$

If there is no queuing phenomenon, the speed at the downstream end of the section is affected by the traffic conditions of the lower section of the section. If there is a vehicle queue phenomenon in the section, the speed of the lower end of the section is determined by the line dissipation rate (i.e., the output capacity). The velocity v_u of the upstream end of the segment is a function of the average density of the segment moving unit:

$$v_u = \begin{cases} v_f, & \rho \leq \rho_{\text{max}}, \\ v_f \left[1 - \left(\frac{\rho - \rho_{\text{max}}}{\rho_{\text{jam}}} \right)^\alpha \right]^\beta, & \rho > \rho_{\text{max}}. \end{cases} \quad (5)$$

2.2.3. Vehicle Moving Model. In this paper, we use the classical model of medium traffic. In the section of the moving part, the speed of the vehicle moving is determined by the speed-density model, which is based on the fixed speed. The deceleration area, if there is no queuing, assumes that beginning t from 0, the vehicle is located in z_0 ; the vehicle at the time of $t(z)$ is defined as

$$t(z) = \frac{z - z_0}{v_u}. \quad (6)$$

Assume that the time is $t = 0$, the vehicle is located in z_0 , and then the position is

$$z(t) = v_u t + z_0. \quad (7)$$

If there is a queue at the location $z(t)$, then the location of the vehicle at any time t is given by the following formula:

$$z(t) = e^{\lambda(t)t} \left(z_0 + \frac{v_u}{\lambda(t)} \right) - \frac{v_u}{\lambda(t)}, \quad (8)$$

$$\text{Where } \lambda(t) = \frac{-v_u}{q_0 + l(ct - m)}.$$

2.3. Data Calibration Algorithm Based on RBF Neural Network. RBF neural network is the radial basis function (RBF) as a hidden unit “base,” constituting a hidden layer, hidden layer to the input vector transform, which can realize the nonlinear mapping from input to output. From the geometric sense, it is equivalent to the given sparse samples point to restore a continuous hypersurface to the point; in the surface to meet the sample value, the network application is equivalent to the estimation of the unknown point value. The learning methods are as follows [14, 15].

Defining the network training sample input as X_k , the actual output of the j output neurons is

$$y_{kj}(X_k) = \sum_{i=1}^N w_{ij} \varphi(X_k, X_i), \quad j = 1, 2, \dots, J. \quad (9)$$

Among them, the Green function $\varphi(X_k, X_i) = G(X_k, X_i)$ is commonly chosen as the basic function; the objective function can be defined as $E = (1/2) \sum_{k=1}^N e_k^2$, N is the number of training samples, and e_k is the signal error, which is defined as $e_k = d_k - Y_k(X_k) = d_k - \sum_{i=1}^I w_i G(\|X_k - t_i(n)\|_{C_i})$.

3. Congestion Control Strategy

3.1. Basic Hypotheses. In the two-way traffic network, the segments are subdivided into segments (Figure 4), and the segments have the same physical characteristics. Each section of the lower reaches its capacity constraints, known as the output capacity. The size of the output power depends on the physical characteristics of the road (such as the width and the slope) or unexpected events, control facilities, and so forth. The section upstream is a mixed traffic area, the area is mixed with the vehicles in each direction of traffic flow, the lower reaches of the area, and the vehicle runs in parallel and independent of each other. The drainage area is subdivided into three separate subregions, which are used for turning, left, straight, and right. In view of the path selection, traffic demand, road characteristics, and so on, this paper makes the following assumptions:

- (1) The traffic between any OD pairs is constant. All paths are obtained by the literature, and they will not change their travel path because of the congestion, unless the traffic control is implemented.
- (2) In the entire simulation time period, it uses the periodic boundary conditions.
- (3) All intersections are defined as having the same input and output power, without any signal control, which only considers the deterministic component of the traffic flow.
- (4) If a particular outlet (e.g., left) is congested, the traffic flow will be formed in the rear of the road section. If obstacles cannot be cleared in time, the line will eventually extend and cross over to the other lanes, causing traffic jams on the road.

3.2. Spatial Structure of Traffic Jam. If a traffic accident occurs in the traffic network, it might as well assume that a road from

east to west in the network will cause a sharp drop in the traffic capacity of the road. When the vehicle line up to the intersection of the road, the plan is to turn left (from south to north) or turn right (north to south) into the road to form a new line of vehicles. Vehicle queue continues to spread until the upstream intersection and causes more road congestion. The distribution of traffic congestion on the network is given after a period of traffic accident. It can be seen that the traffic jam on the space structure is generally a diamond structure. But according to the latest research results show that there is no accident of two-way lattice network congestion space structure which is generally a central symmetrical graphics. After the accident, the traffic jam is extended from the accident to the tree structure, which is shown in the paper. If the accident lasts long enough, the range of Congestion Propagation in four directions to the four corners of the world will be roughly the same. This feature is crucial to ease traffic congestion, because these sections for jam provide a buffer zone.

In order to show the growth process of congestion, Figure 1 depicts the conditions where there is no artificial intervention in the Congestion Propagation tree. Figure 1 (left) shows the process of the accident which removes the previous congestion growth process. It can clearly see the scale of the congestion and the network in which each section of the road is in a congested state of the total time. Figure 1 (right) shows the formation of a closed loop. When the accident lasts long enough, the stop line width is divided into an unreasonable (turning direction allocation), and the closed loop is formed in the process of the traffic accident relief. Once the incident is cleared, the vehicle starts and generates the start wave. At the same time, the upper reaches of the vehicle continue to enter the congested area. At first, a straight line of vehicles can effectively leave the congestion area, while the queue also looks to start to dissipate and then turn the vehicle quickly to get to the end of the initial line and line up [16]. Due to the distribution of the direction of the traffic flow direction of the car park, the traffic flow direction is insufficient, which causes that the turning vehicles cannot leave the congestion area in time. Sometime later, straight line direction of the traffic flow is affected by the turning of the vehicle and fewer vehicles are pulled out of the jam area, while the arrival of regional congestion of vehicles is increasing. This has led to a growing traffic congestion. Without manual intervention, congestion will continue to grow until the entire network is covered.

3.3. New Congestion Control Strategy. In a one-way street grid, the exit of a section of the road is divided into two directions: straight and turning (left or right). In this form of traffic, the vehicle can be used to restrict the area of the vehicle entering the area by using the measures to prohibit the use of a straight line or a turn. In the two-way traffic network, the export of road segments needs to be divided into three directions: turn left, go straight, and turn right. For the same section, not only can the straight line of the vehicle enter the congested area, but also left or right turn vehicle can also enter the congested area. In this form of transportation, the simple

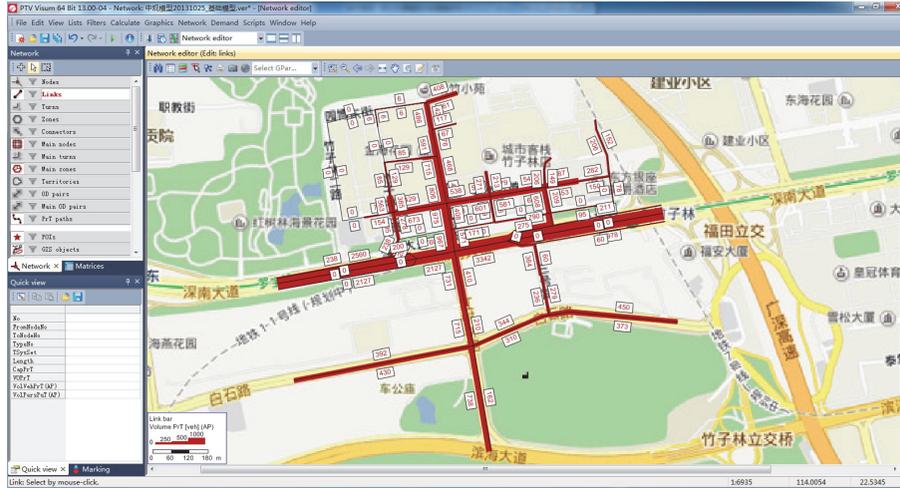


FIGURE 2: Flow distribution of Zhuzilin area roads.

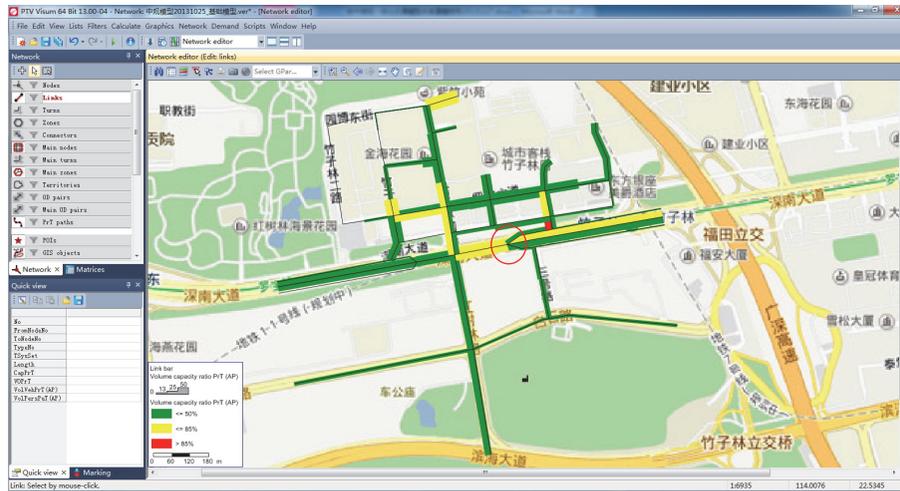


FIGURE 3: Regional road saturation distribution.

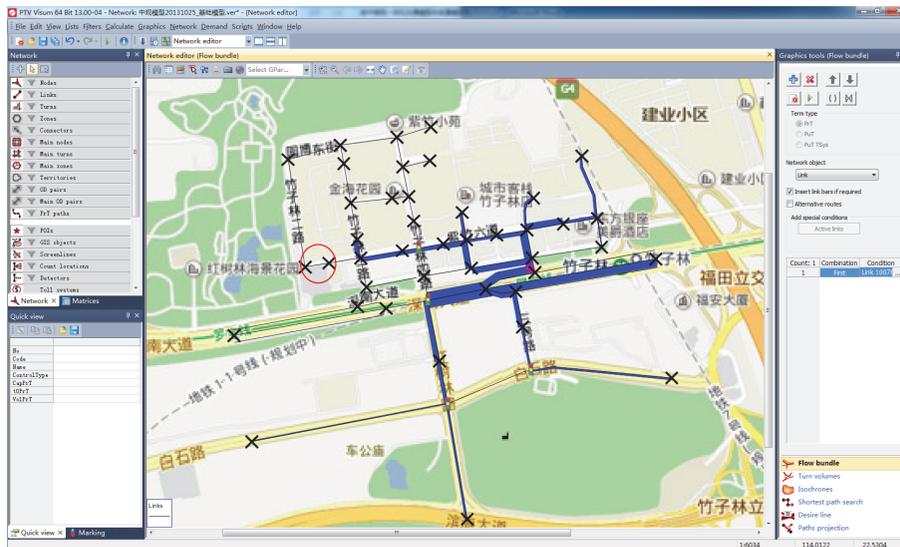


FIGURE 4: Road congestion cobweb diagram.

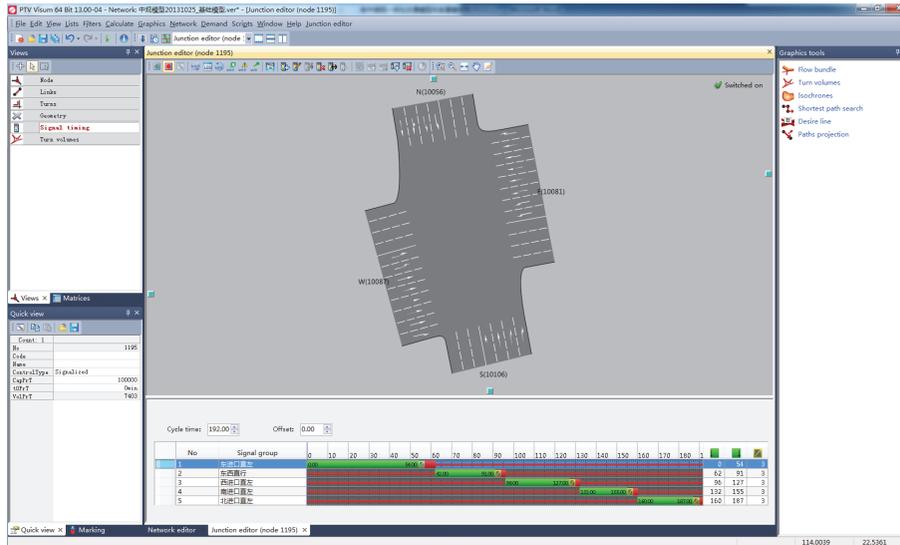


FIGURE 5: Current situation of the intersection of the channel control scheme.

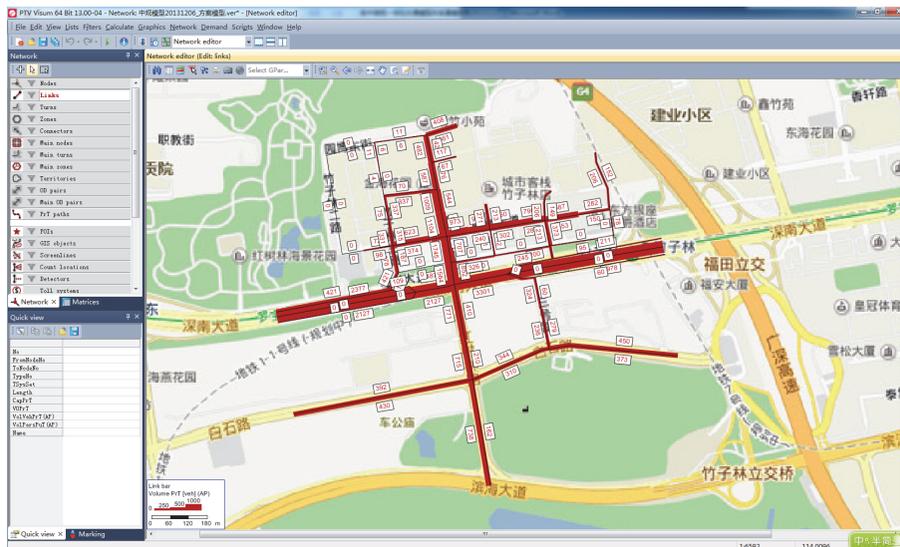


FIGURE 6: Road traffic distribution (Zhuzilin area modify credit control scheme).

use of a ban on the use of direct line of measures cannot effectively limit the congestion area of the vehicle to enter the congested area. Thus, the existing congestion control strategy for a one-way street network can not be completely copied to the two-way traffic network [17].

In this paper, the congestion control strategy is to turn prohibition and the vehicles intelligent behavior, and in the periphery of the congested area a vehicle prohibition area is established, while the vehicle changes its route in congested area to take a temporary path. According to the shape of the congestion, the control strategy is presented as follows: (1) single track control strategy, (2) multiline control strategy, (3) control strategy, and (4) tree control strategy.

The fourth kind of control strategy is proposed in this paper that the formation of congestion is not only concerned

with the site of the incident and traffic network structure and other factors are inseparable, but two-way lattice network, Congestion Propagation, should be in the path that the accident site is in the OD of the shortest path, whose shape is similar to the tree branches. So it is necessary to execute a variety of intelligent or artificial control measures in the periphery of the structure of congestion; hence the name is the tree control strategy [12, 18–20]. Of course, this is no longer a single restriction; there is also induction of real-time path selection behavior. In the medium traffic model, the measures are forbidden to turn and real-time route choice in order to ease the traffic congestion. In the paper, the vehicle is used to cancel the condition that if the traffic congestion at the lower reaches the accident site which is completely dissipated, all the measures will be cancelled immediately.

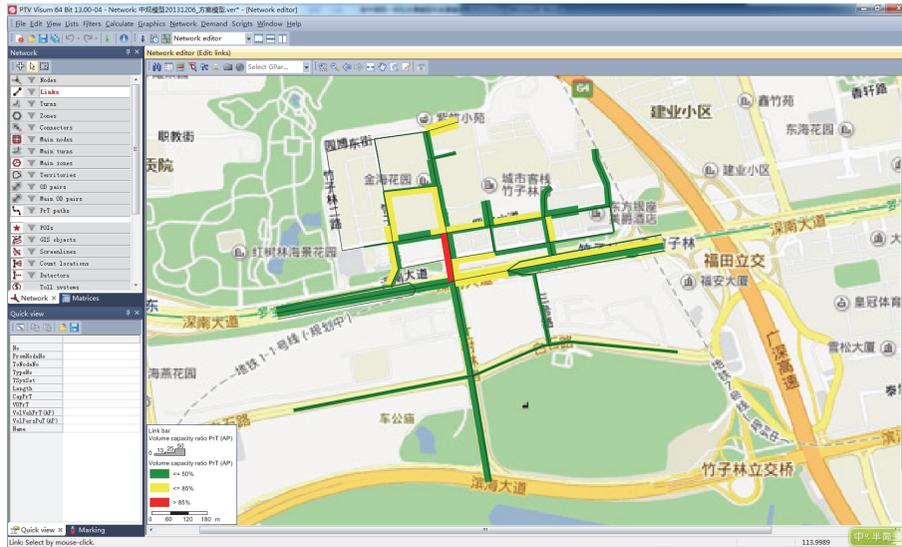


FIGURE 7: Road traffic distribution (Zhuzilin area modify credit control scheme).

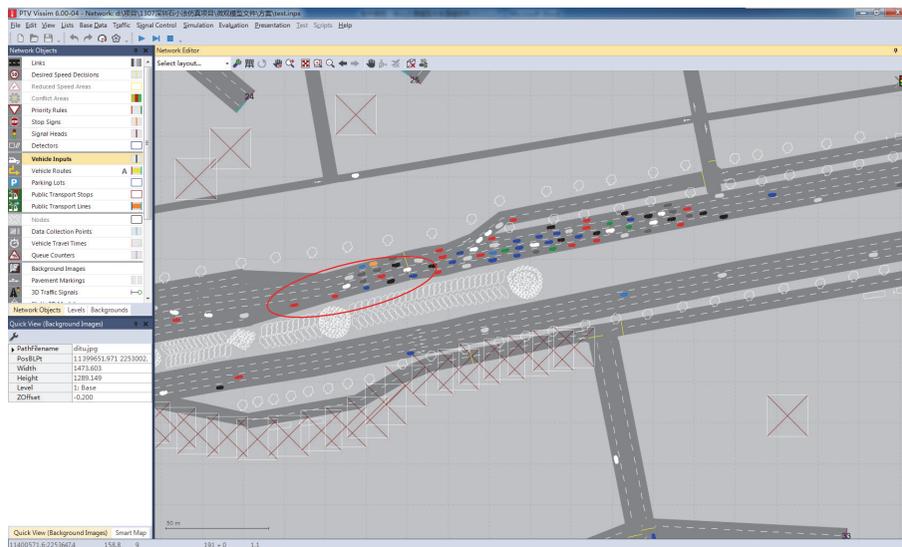


FIGURE 8: Vehicle queuing situation diagram.

4. Case Study

In this paper, we observe the average density flow rate and other indicators in the above conditions, so as to judge the congestion of the network. Secondly, the OD is set up, the OD points are defined, and the fourth kind of congestion dissipation strategies are adopted. Average density flow rate and other indicators are then analyzed. The accident occurred at the beginning of the accident upstream, if the entire simulation time is not to take any measures, then the queue continues to spread. According to the concept of traffic model, the traffic flow and saturation Zhuzilin area distribution are as shown below. From the perspective of traffic demand and traffic supply, the medium traffic model is to analyze the causes of traffic congestion and to analyze the

congestion control scheme. In this study, the Zhuzilin area is used to establish an example of a mesoscopic traffic model for congestion causes analysis and to make recommendations for improvement. Because this project is mainly integrated model of the technical analysis method research, the results may be different from the actual traffic conditions.

4.1. Cause Analysis of Congestion. According to the concept of traffic model, the traffic flow and saturation Zhuzilin area distribution are shown in Figures 2–6.

As can be seen from Figure 3, the Zhuzilin area is the main point of congestion on Zhuzilin Road, specifically, the six roads leading to the Shennan Boulevard Voeux Road (circled in red) sections. Using the Visum flow bundle function, the view of the road traffic situation is shown in Figure 4.

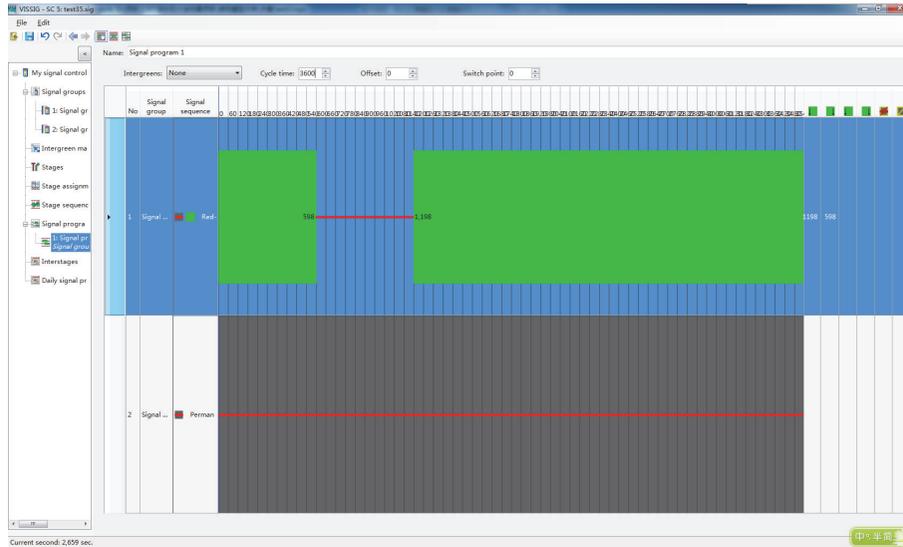


FIGURE 9: Signal lamp setup diagram.

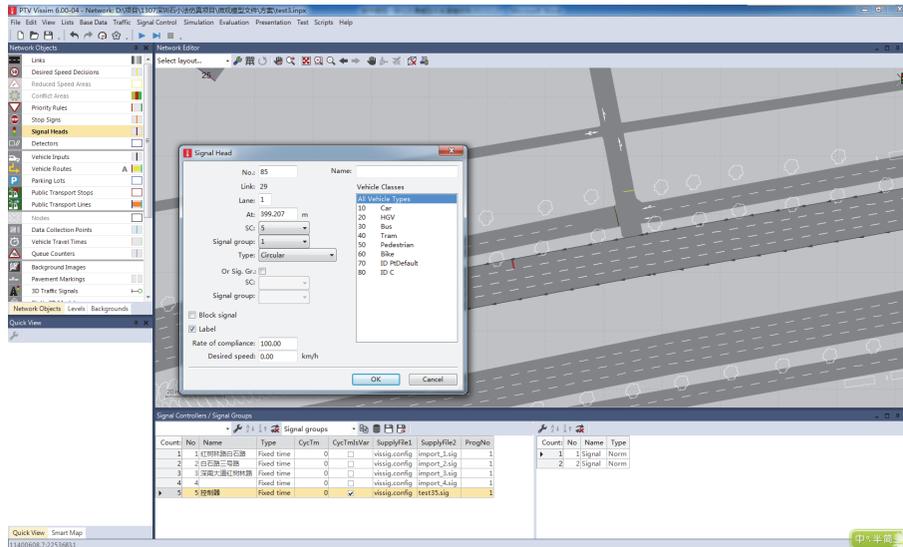


FIGURE 10: Signal lamp setup diagram.

The spider web analysis diagram shows that this section of the traffic is mainly to the Shennan Road East of the U-turn car flow, which has a part of the traffic flow or through the northwest corner of the intersection of Shennan Avenue-Mangrove Road (pictured in red circle location) leading to the Shennan Road East. This part of the car flow is from four Zhuzilin Road left turn lanes into Shennan Avenue. We use flow bundle function and check the traffic situation of the four Zhuzilin Road left turn lanes into Shennan Avenue.

It is found that the traffic from the four Zhuzilin left turn lanes into Shennan Road traffic is large. Therefore, the Shennan Avenue-Mangrove Road intersection signal control and channelization schemes were studied. It is found that the north entrance of the left turn traffic green signal ratio is only

0.14. According to the investigation of the saturation current of the intersection, the distance of the head is 2.53 seconds; that is, the saturation capacity of single lane is 1400 pcu/h. From this calculation, the actual turning capacity is $2.5 * 1400 * 0.14 = 490$ pcu/h. At the same time, Shennan Road East imports the leftmost lane for left U-turn lane (due to the presence of two dedicated left turn lanes, the survey found that the lane is 80% of the traffic to U-turn car flow), and the green signal ratio is 0.28, which is calculated to get the actual turnover capacity for $0.8 * 1400 * 0.28 = 310$ pcu/h. When the volume from the four Zhuzilin left turn lanes into Shennan Road traffic is excessive, part of the traffic flow opts to reach Zhuzilin Road via the six lanes of Shennan Avenue after making a U-turn after traveling east, which is shown in Figure 5.

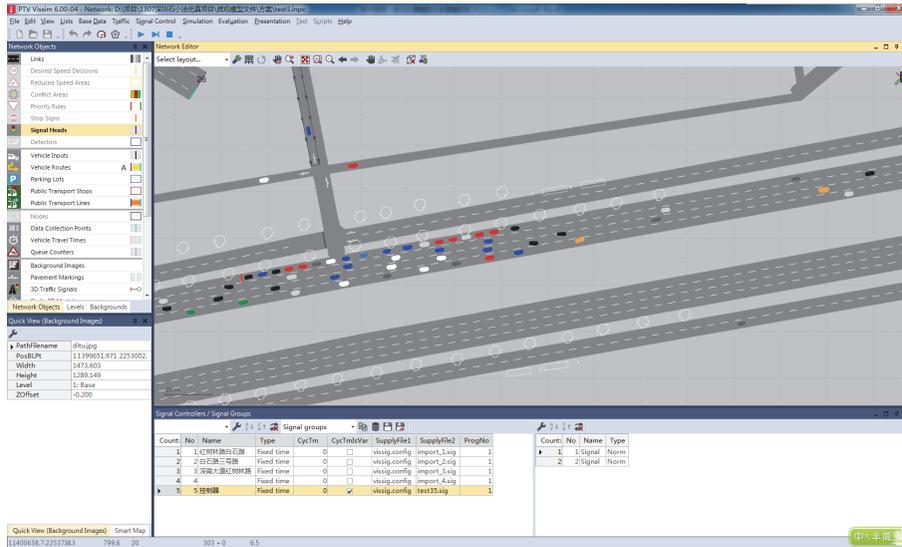


FIGURE 11: Queue start chart.

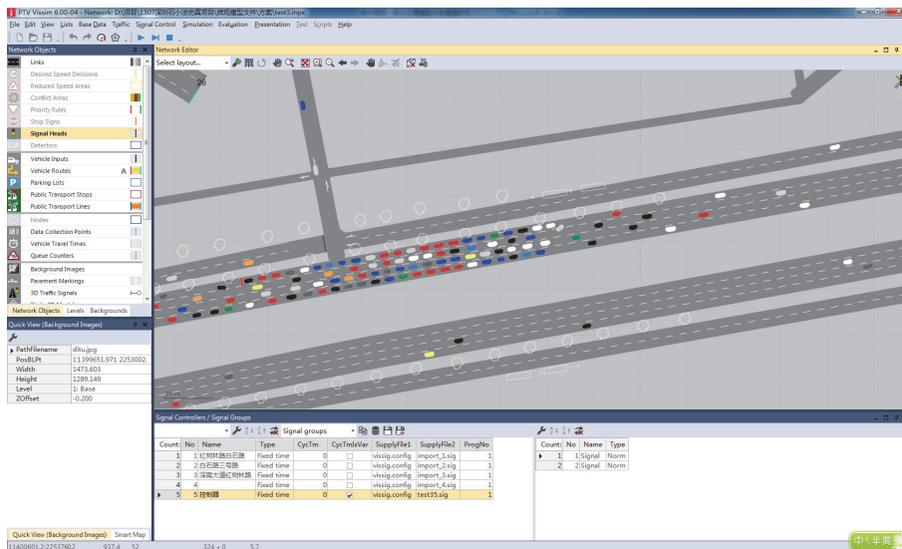


FIGURE 12: Queuing growth process.

Thus, the main cause being the four Zhuzilin left lanes overloading Shennan Road's traffic capacity is insufficient. According to the above reasons, to improve the design of Zhuzilin four Shennan Road, it is necessary to increase the capacity of left turn including the increase of the steering drive and improving the green letter. Considering that there are already 2.5 left turn lanes, the proposed method is proposed to improve the green letter ratio. Also, the fourth control strategy is used.

Thus, the calculated regional traffic flow and saturation distribution are shown as Figures 6 and 7.

It can be seen from Figure 6 and 7 that the improved scheme can be significantly decreased by routing six lanes of traffic from Zhuzilin Road into Shennan Boulevard des Voeux Road traffic; the traffic will choose four Zhuzilin

Road left turn lanes into Shennan Avenue. At the same time, however, we also found that the congestion of the four Zhuzilin Road turn lanes increased, which requires further analysis, which we have deferred.

4.2. *Microscopic Traffic Model.* Microscopic traffic model is based on the driving of vehicles in the road network and analyzes the bottleneck of the traffic congestion, and the improvement of the bottleneck points is analyzed [14]. In this section, for project established using the Zhuzilin area microscopic traffic model, for example, the congestion bottleneck analysis and process improvement measures are presented. Because this project is mainly integrated model of the technical analysis method research, the results may be different from the actual traffic conditions.

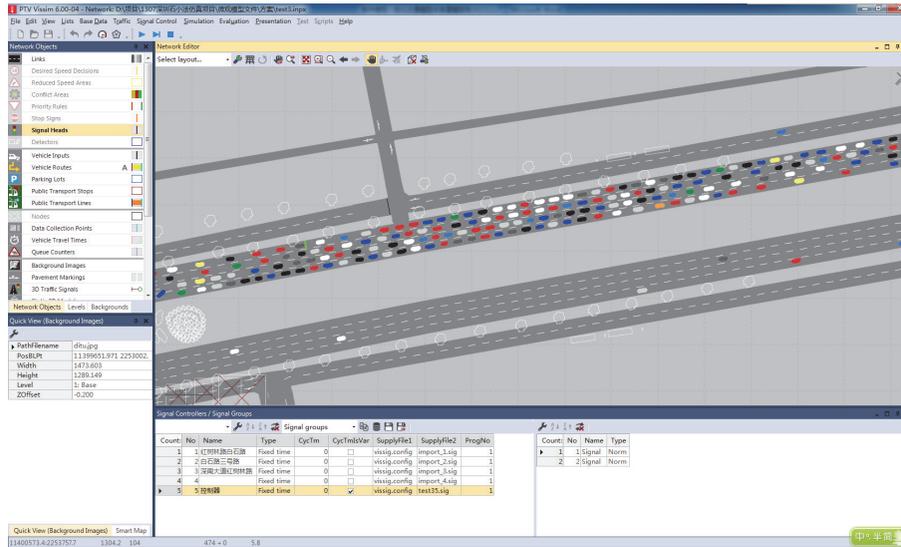


FIGURE 13: Accident handling line began to evacuate.

As shown in the Figure 8, the location of the congestion bottleneck through the vehicle queue is determined.

Analysis of the queuing situation is due to the formation of the main road, and the middle section is the bottleneck of the whole road traffic. Microscopic traffic model can be used to analyze the causes of sudden traffic congestion and traffic congestion, to analyze the feasibility of the scheme, and to evaluate the effect of different traffic congestion on the traffic flow. The simulation of the emergency traffic congestion is analyzed. The microscopic traffic model analysis method used in the traffic congestion is similar to that used in the analysis of the traffic congestion. Therefore, in this section, for the project established using the Zhuzilin area microscopic traffic model, for example, the congestion response speed and evacuation analysis process are discussed.

Because this project is mainly integrated model of the technical analysis method research, the results may be different from the actual traffic conditions. In the microscopic model, an accident caused by a road bypass signal lamp is simulated. For example, we have to take 10 minutes to complete the accident, and the accident occurred in the tenth minutes. The signal light can be then synchronized with the red light for about 600 s, until about 1200 s, when it is synchronized with the green light, as shown in Figures 9–13.

5. Conclusion

Traffic incident is one of the main reasons leading to the traffic jam. In this paper, the traffic congestion caused by accidents is simulated by using the improved medium traffic model, and the propagation law of traffic congestion is found. The control strategy is put forward for the first time. The simulation results also prove that the control strategy proposed in this paper is effective and feasible. According to the temporal and spatial evolution of the paper, we can see that the network has been improved on the whole. But the paper

is still showing many irregularities, such as not considering the use of advanced intelligent transportation methods for congestion control, cross-signaling intersection control, and information guidance. Also not considered was the use of more advanced traffic prediction methods, informed in advance by dynamic network information, and antiblocking, as well as the effective combination of the above-mentioned methods. This is also the direction of future efforts of the author.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the Guangdong Natural Science Fund (2015A030313626), Special Fund of Frontier and Key Technology Innovation of Guangdong Province (Provincial Major Science and Technology Project) (2014B010112008, 2015B010106001, 2015B010129003, and 2016B010109007), Pearl River S&T Nova Program of Guangzhou (201610010034), Guangdong Science and Technology Program (2013B01010201, 2013B010102018, and 2013B090600152), Guangzhou Science and Technology Project (no. 2014Y2-00044), National Natural Science Foundation of China (61603105), and Scientific Research Project of Guangzhou Municipal Colleges and Universities (1201610154).

References

- [1] M. Junghare and M. Shimpi, "A survey of congestion control protocols for wireless sensor network," in *Proceedings of the IJCA Proceedings on National Conference*, vol. 43, 2012.

- [2] J. Wan, X. Xu, R. Feng, and Y. Wu, "Cross-layer active predictive congestion control protocol for wireless sensor networks," *Sensors*, vol. 9, no. 10, pp. 8278–8310, 2009.
- [3] O. Jahn, R. H. Möhring, A. S. Schulz, and N. E. Stier-Moses, "System-optimal routing of traffic flows with user constraints in networks with congestion," *Operations Research*, vol. 53, no. 4, pp. 600–616, 2005.
- [4] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2125–2136, 2017.
- [5] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [6] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.
- [7] Z. Zhao, Y. Liu, and F. Luo, "Output feedback boundary control of an axially moving system with input saturation constraint," *ISA Transactions*, vol. 68, pp. 22–32, 2017.
- [8] Z. Zhao, Y. Liu, F. Guo, and Y. Fu, "Vibration control and boundary tension constraint of an axially moving string system," *Nonlinear Dynamics*, vol. 89, no. 4, pp. 2431–2440, 2017.
- [9] Z. Zhao, Y. Liu, and F. Luo, "Boundary control for a vibrating string system with bounded input," *Asian Journal of Control*, In press.
- [10] Z. Zhao, Y. Liu, W. He, and F. Luo, "Adaptive boundary control of an axially moving belt system with high acceleration/deceleration," *IET Control Theory & Applications*, vol. 10, no. 11, pp. 1299–1306, 2016.
- [11] Z.-J. Zhao, Y. Liu, F. Guo, and Y. Fu, "Modelling and control for a class of axially moving nonuniform system," *International Journal of Systems Science*, vol. 48, no. 4, pp. 849–861, 2017.
- [12] W. K. Newey, "Nonparametric continuous/discrete choice models," *International Economic Review*, vol. 48, no. 4, pp. 1429–1439, 2007.
- [13] S. Chen and N. Yang, "Congestion avoidance based on lightweight buffer management in sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 17, no. 9, pp. 934–946, 2006.
- [14] D. Howard, B. Mark, and H. Martin, *Neural Network Toolbox For Use with Matlab*, The MathWorks, Natick, Mass, USA, 2007.
- [15] N. F. Zhang, J. F. Yang, Y. J. Xue, Z. Li, and X. L. Huang, "Agricultural machinery operation posture rapid detection intelligent sensor calibration method based on RBF neural network," *Applied Mechanics and Materials*, vol. 373–375, pp. 932–935, 2013.
- [16] A. Petrin, "Quantifying the benefits of new products: the case of the minivan," *Journal of Political Economy*, vol. 110, no. 4, pp. 705–729, 2002.
- [17] S. Peeta and A. K. Ziliaskopoulos, "Foundations of dynamic traffic assignment: the past, the present and the future," *Networks and Spatial Economics*, vol. 1, no. 3–4, pp. 233–265, 2001.
- [18] Z. Zhao, X. Wang, C. Zhang, Z. Liu, and J. Yang, "Neural network based boundary control of a vibrating string system with input deadzone," *Neurocomputing*, 2017.
- [19] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.
- [20] W. He, Y. Dong, and C. Sun, "Adaptive neural impedance control of a robotic manipulator with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334–344, 2016.

Research Article

Fault Diagnosis Method of Check Valve Based on Multikernel Cost-Sensitive Extreme Learning Machine

Jun Ma,¹ Jiande Wu,^{2,3} and Xiaodong Wang^{2,3}

¹Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming 650500, China

²Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China

³Engineering Research Center for Mineral Pipeline Transportation YN, Kunming 650500, China

Correspondence should be addressed to Jiande Wu; wjiande@kmust.edu.cn

Received 7 July 2017; Accepted 8 November 2017; Published 28 December 2017

Academic Editor: Yanan Li

Copyright © 2017 Jun Ma et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Check valve is one of the most important components and most easily damaged parts in high pressure diaphragm pump, which is a typical representative of reciprocating machinery. In order to ensure the normal operation of the pump, it is necessary to monitor its running state and diagnose fault. However, in the fault diagnosis of check valve, the classification models with single kernel function can not fully interpret the classification decision function, and meanwhile unreasonable assumption of diagnostic cost equalization has a significant impact on classification results. Therefore, the multikernel function and cost-sensitive mechanism are introduced to construct the fault diagnosis model of check valve based on the multikernel cost-sensitive extreme learning machine (MKL-CS-ELM) in this paper. The comparative test results of check valve for high pressure diaphragm pump show that MKL-CS-ELM can obtain fairly or slightly better performance than ELM, CS-ELM, MKL-ELM, and multikernel cost-sensitive support vector learning machine (MKL-CS-SVM). At the same time, the presented method can obtain very high accuracy under imbalance datasets condition and effectively overcome the weakness of diagnostic cost equalization and improve the interpretability and reliability of the decision function of classification model. It, therefore, is more suitable for the practical application.

1. Introduction

High pressure diaphragm pump is the most important equipment for high concentration slurry pipeline transportation. Its working condition is directly related to whether the pump can be restarted after stopping and whether it will produce accelerated flow in batch transportation. Check valve is the core and the easiest damaged component of the high pressure diaphragm pump. In order to ensure the normal operation of the pump, it is necessary to monitor its running state and diagnose fault [1]. So, the research of condition monitoring and fault diagnosis of the high pressure diaphragm pump has important practical significance in promoting development of slurry pipeline transportation field.

However, the fault characteristics of reciprocating machinery are difficult to extract because of its complex structure, multiple excitation sources, unstable operation, and so on [2]. In order to complete the condition monitoring

and fault diagnosis of reciprocating machineries effectively, both domestic and foreign scholars have introduced the fault diagnosis methods of rotating machinery into the fault diagnosis of reciprocating machinery and made many valuable research results [3–5]. Ogle and Morrison [6] analyzed the failure accident of diaphragm pump and found that the environmental stress cracking of diaphragm is one of the main reasons for the diaphragm pump failure. The research results have provided effective theoretical support for accident prevention and pipeline maintenance and greatly reduced maintenance costs. In recent years, wavelet transform and Fourier transform, information entropy, neural network, bispectrum analysis, feature fusion, evidence theory, chaos theory, fractal theory, decision tree, and SVM have been widely applied to the fault diagnosis of reciprocating machinery, and many significant research achievements have been obtained [7–16]. Yet compared with the fault diagnosis of rotating machinery, there are still many research

contents to be improved: (1) the data sample size of reciprocating machinery is huge and a great deal of multisource heterogeneous information is held within them due to the influence of complex structure, multiple excitation source, multiple wearing parts, coupling of the signal, and strong nonlinearity of reciprocating machinery. It is not reasonable to use a single kernel function (such as radial basis function kernel, polynomial kernel function) for processing all the samples, and it is unable to explain the signal completely. Consequently, it is an inevitable choice to combine multiple kernel functions to achieve better processing results [17–19]. (2) It is impossible for fault diagnosis models to get ideal classification results when datasets of fault diagnosis are not balanced (the fault samples are far less than the normal samples) and the diagnostic cost is unequal (e.g., the diagnostic cost between “the normal state which is identified as a fault state” and “the fault state which is identified as a normal state” is quite different; the former will only result in an “invalid” examining and repair for operator, but the latter will result in major safety incidents), so the hypothesis deficiency of minimum classification error and diagnostic cost equalization in the existing classification model need to be overcome [20]. (3) At present, BP neural network and SVM are relatively mature classification learning methods and play important role in the fault diagnosis of reciprocating machinery. But, BP neural network has the problems of easily falling into local minimum, being not convergence, and so on. Meanwhile, the optimization calculation load of SVM increases with the optimization parameters and data sample size. And many parameters will be optimized to get the optimization SVM classification model. It, therefore, is one of the hot topics to explore new classification method which has the advantages of fast training speed and fewer optimization parameters to obtain global optimal solution [21].

In recent years, ELM is widely used because of its effectiveness, high speed, being easy for implementation, and multiclassification in the related fields of machine learning [22–24]. Moreover, the modified ELM models can validly solve the problems of imbalance sample and obtain better performance [25–28]. Therefore, the modified ELM methods have become the main research direction. For one thing, the transfer function of the original hidden layer based on random feature mapping will be substituted for the more efficient transfer functions. Then, the sigmoid function and radial basis function (RBF) [29–32] which are widely used in neural networks have been introduced into ELM and obtained better experiments results. For another thing, how to improve classification performance of ELM under multisource heterogeneous data and information fusion is also one of the latest research trends of the modified ELM classification models. Liu et al. [33] proposed the multikernel ELM (MKL-ELM) combined with the multikernel learning with ℓ_p constraints. Compared with traditional ELM, the MKL-ELM can solve these issues, including the selection and optimization of multikernel function, the application of multisource heterogeneous data processing method, and information fusion method in the classification. But, in [33], the researcher does not consider the impact of classification cost on the classification model. So, the cost-sensitive mechanism

was introduced into the conventional ELM [34], and a new classification model based on cost-sensitive is proposed to conquer the drawback of diagnostic cost equalization. But, it is not very effective in dealing with the multisource heterogeneous data and information fusion because of the restriction of single and permanent kernel during the subsequent processing.

With the intensive study of ELM theory and application, the MKL-ELM and CS-ELM have greatly promoted the development of ELM. But there is still plenty of room for improvement and extension. This is typically shown in two aspects: (1) how to select the most appropriate cost-sensitive method; (2) how to construct more general multikernel function which can be widely used in fault diagnosis field. Based on the points discussed above, the multikernel function and cost-sensitive mechanism are introduced into ELM to construct the fault diagnosis model based on MKL-CS-ELM for check valve of high pressure diaphragm pump in this paper.

This paper has the following main contributions. First, the advantages, shortcomings, and the application ranges of oversampling, undersampling, and threshold adjusting are analyzed to provide theoretical support for the choice of cost-sensitive methods. Second, a new fault diagnosis method based on MKL-CS-ELM is proposed to diagnose the check valve faults of high pressure diaphragm pump. Third, the comparison experiments of ELM, CS-ELM, MKL-ELM, MKL-CS-SVM, and MKL-CS-ELM are carried out, and the effectiveness of the proposed MKL-CS-ELM method is verified.

The remainder of this paper is organized as follows. Section 2 describes the fundamental theory of ELM, MKL-ELM, cost-sensitive learning, and evaluation index of classification model. Section 3 presents the implementation process of the proposed method in detail. Section 4 elaborates experimental process. Section 5 shows the experimental results analysis. Section 6 offers the discussion and conclusion.

2. Related Work

2.1. Extreme Learning Machine (ELM). From the classification optimization point of view, the principle of ELM is similar to SVM and LSSVM, whose goal is to obtain the minimum training error and maximum classification margin or generalization ability. So, on the basis of SVM principle analysis, the optimized mathematical model of ELM is described as follows [35]:

$$\begin{aligned} \min_{\beta, \xi} \quad & \frac{1}{2} \|\beta\|_F^2 + \frac{C}{2} \|\xi\| \\ \text{s.t.} \quad & \beta' \phi(x_i) = y_i - \xi_i, \quad \forall i. \end{aligned} \quad (1)$$

In (1), $\beta \in \mathfrak{R}^{|\phi(\cdot) \times T|}$ stands for the connecting weighting coefficients between hidden layer and output layer, $\|\cdot\|_F$ is Frobenius norm, C represents regularization parameter or penalty factor which achieves the balance between the minimum training error and maximum classification margin, $\xi_i = [\xi_{1i}, \xi_{2i}, \dots, \xi_{Ti}]'$, ($1 \leq i \leq n$) is the i th column of the error matrix $\xi \in \mathfrak{R}^{T \times n}$, β' represents the transpose of

matrix β (similarly hereinafter), $\phi(x_i)$ is the output function of hidden layer for the input neuron x_i , $\{(x_i, y_i)\}_{i=1}^n$ represents the given training set, and $y_i = [0, \dots, 0, 1, 0, \dots, 0]^t \in \{0, 1\}^l$ represents the case that sample x_i belongs to the classification label t . n and T are the number of training samples and categories, respectively. According to KKT (Karush Kuhn Tucker) theory, the analysis solution of (1) is calculated and the detailed solution process can be read in [36]. The solution of the output weight β^* is solved using the Moore-Penrose $\widehat{\Phi}^+$:

$$\begin{aligned} \beta^* &= \widehat{\Phi}^+ Y = \Phi^T \left(\frac{I}{C} + \Phi \Phi^T \right)^{-1} Y' \quad n < l, \\ \beta^* &= \widehat{\Phi}^+ Y = \left(\frac{I}{C} + \Phi \Phi^T \right)^{-1} \Phi^T Y' \quad l < n. \end{aligned} \quad (2)$$

In (2), the output matrix of the hidden layer is $\Phi = [\phi(X_1), \dots, \phi(X_n)]^t \in \mathfrak{R}^{n \times |\phi(\cdot)|}$, the output result of ELM classification model is $Y = [y_1, \dots, y_n] \in \mathfrak{R}^{T \times n}$, and I is identity matrix.

For a given new sample x , the output decision function $f(x)$ of the ELM is shown as follows:

$$f(x) = \beta^{*t} \phi(x). \quad (3)$$

2.2. Multikernel Extreme Learning Machine (MKL-ELM). The common definition of multikernel function is the linear combination of basic kernel function. So, the combination coefficient of optimal kernel function and the maximum margin of ELM are the key and core of the MKL-ELM [33]. A typical form of multikernel function is shown in

$$\kappa(\cdot, \cdot; \gamma) = \sum_{p=1}^m \gamma_p \kappa_p(\cdot, \cdot). \quad (4)$$

In (4), $\{\kappa_p(\cdot, \cdot)_{p=1}^m\}$ represents m basic kernel functions.

For the convenience of processing and computing, the combination coefficients γ_p of basic kernel function satisfy restricting condition $\sum_{p=1}^m \gamma_p = 1$. The feature mapping of (4) is shown in

$$\phi(\cdot; \gamma) = [\sqrt{\gamma_1} \phi_1(\cdot), \sqrt{\gamma_2} \phi_2(\cdot), \dots, \sqrt{\gamma_m} \phi_m(\cdot)]. \quad (5)$$

In (5), $\phi(\cdot, \gamma)$ and $\{\phi(\cdot)_{p=1}^m\}$ are the high dimensional feature mapping of $\kappa(\cdot, \cdot; \gamma)$ and $\{\kappa_p(\cdot, \cdot)_{p=1}^m\}$, respectively.

In the construction process of multikernel function, RBF kernel function, Laplace kernel function, and inverse-distance kernel function are selected as basic kernel functions.

RBF kernel function: $k(x_i, x_j)$

$$= \exp\left(\frac{-\|x_i - x_j\|^2}{\sigma}\right)$$

Laplace kernel function: $k(x_i, x_j)$

$$= \exp\left(\frac{-\|x_i - x_j\|}{\sqrt{\sigma}}\right)$$

Inverse-distance kernel function: $k(x_i, x_j)$

$$= \frac{1}{(\|x_i - x_j\| / \sqrt{\sigma} + 1)}.$$

(6)

In (6), σ is the parameter of kernel function. In this paper, the value of σ is calculated by $\sigma = 2^t \sigma_0$ ($t \in \{-2, -1, 0, 1, 2\}$). At the same time, σ_0 represents the mean Euclidean distance between the samples.

In order to insure that the final solution and combination kernel function of multikernel optimal problem are subject to the boundedness and symmetric positive semidefinite, respectively, the ℓ_q norm is used as the constraint condition of the combination coefficient γ of the multikernel function. The different value of q in the ℓ_q norm represents different constraint norm. According to the theoretical basis of multikernel SVM [37, 38] and (5), the theoretical expression of conventional MKL-ELM is described as follows:

$$\begin{aligned} \min_{\gamma} \min_{\beta, \xi} \quad & \frac{1}{2} \|\beta\|_F^2 + \frac{C}{2} \sum_{i=1}^n \|\xi_i\|^2 \\ \text{s.t.} \quad & \beta^t \phi(x_i; \gamma) = y_i - \xi_i, \quad \forall i = 1, \dots, n, \\ & \sum_{p=1}^m \gamma_p^q = 1, \quad \gamma_p \geq 0, \quad \forall p = 1, \dots, m. \end{aligned} \quad (7)$$

In (7), the connecting weighting coefficient β is $\beta = [\beta_1, \dots, \beta_m] \in \mathfrak{R}^{(|\phi_1(\cdot)| + \dots + |\phi_m(\cdot)|) \times T}$ and $\beta_p \in \mathfrak{R}^{(|\phi_p(\cdot)|) \times T}$ ($p = 1, \dots, m$) is the connecting weighting of the p th basic kernel function.

Substituting (5) into (7), the expression of MKL-ELM is obtained and shown in

$$\begin{aligned} \min_{\gamma} \min_{\beta, \xi} \quad & \frac{1}{2} \|\beta\|_F^2 + \frac{C}{2} \sum_{i=1}^n \|\xi_i\|^2 \\ \text{s.t.} \quad & \sum_{p=1}^m \sqrt{\gamma_p} \beta_p^t \phi_p(x_i) = y_i - \xi_i, \quad \forall i = 1, \dots, n, \\ & \sum_{p=1}^m \gamma_p^q = 1, \quad \gamma_p \geq 0, \quad \forall p = 1, \dots, m. \end{aligned} \quad (8)$$

If $\tilde{\beta} = [\tilde{\beta}_1, \tilde{\beta}_2, \dots, \tilde{\beta}_m]$ is equal to $[\sqrt{\gamma_1} \beta_1, \sqrt{\gamma_2} \beta_2, \dots, \sqrt{\gamma_m} \beta_m]$, then (8) can be simplified to

$$\begin{aligned} \min_{\gamma} \min_{\beta, \xi} \quad & \frac{1}{2} \sum_{p=1}^m \frac{\|\tilde{\beta}_p\|_F^2}{\gamma_p} + \frac{C}{2} \sum_{i=1}^n \|\xi_i\|^2 \\ \text{s.t.} \quad & \sum_{p=1}^m \tilde{\beta}_p^t \phi_p(x_i) = y_i - \xi_i, \quad \forall i = 1, \dots, N, \\ & \sum_{p=1}^m \gamma_p^q = 1, \quad \gamma_p \geq 0, \quad \forall p = 1, \dots, m. \end{aligned} \quad (9)$$

Equation (9) is similar to the expression of ELM. So, the Lagrangian function of MKL-ELM can be calculated:

$$L(\tilde{\beta}, \xi, \gamma) = \frac{1}{2} \sum_{p=1}^m \frac{\|\tilde{\beta}_p\|_F^2}{\gamma_p^q} + \frac{C}{2} \sum_{i=1}^n \|\xi_i\|^2 - \sum_{t=1}^T \sum_{i=1}^n \alpha_{it} \left(\sum_{p=1}^m \tilde{\beta}_p' \phi_p(x_i) - y_{ti} - \xi_{ti} \right) + \tau \left(\sum_{p=1}^m \gamma_p^q - 1 \right). \quad (10)$$

In (10), $\alpha \in \mathfrak{R}^{n \times T}$ and τ are the Lagrangian multiplier. Then, KKT optimization condition is calculated and shown in

$$\tilde{\beta}_p = \gamma_p^q \sum_{t=1}^T \sum_{i=1}^n \alpha_{it} \phi_p(x_i), \quad \forall p, \quad \xi_{ti} = \frac{\alpha_{ti}}{C}, \quad \forall t, \forall i, \quad (11)$$

$$\sum_{p=1}^m \tilde{\beta}_p' \phi_p(x_i) = y_i - \xi_i.$$

The matrix form of (11) is expressed in

$$\left(\kappa(\cdot, \cdot; \gamma) + \frac{I}{C} \right) \alpha = Y'. \quad (12)$$

In (12), the compound kernel function $\kappa(\cdot, \cdot; \gamma)$ represents $\kappa(x_i, x_j; \gamma) = \phi(x_i; \gamma)' \phi(x_j; \gamma) = \sum_{p=1}^m \gamma_p K(x_i, x_j)$. Then, the solution of α is shown as follows:

$$\alpha = \left(\kappa(\cdot, \cdot; \gamma) + \frac{I}{C} \right)^{-1} Y'. \quad (13)$$

At the same time, the combination coefficient γ of multi-kernel function can be calculated by the derivative of

$$\gamma_p^{\text{new}} = \frac{\|\tilde{\beta}_p\|_F^{2/(1+q)}}{\left(\sum_{p=1}^m \|\tilde{\beta}_p\|_F^{2q/(1+q)} \right)^{1/q}}, \quad \forall p. \quad (14)$$

The sparse MKL-ELM constrained ℓ_1 norm is given by $q = 1$ in (14). The optimal parameter of α^* and γ^* is calculated by iterative optimization methods. Now, for a given sample x , the output decision function $f(x)$ of MKL-ELM is expressed as follows:

$$f(x) = [f_1(x), \dots, f_T(x)]. \quad (15)$$

In (15), the component of $f(x)$ represents $f_c(x) = \sum_{i=1}^n \alpha_{ic}^* \sum_{p=1}^m K_p(x_i, x)$, $1 \leq c \leq T$.

2.3. Cost-Sensitive Methods. The cost-sensitive methods largely fall into three groups [39]: constructing the cost-sensitive classification model directly, establishing the cost-sensitive classification model using the Bayesian risk theory, and building the cost-sensitive classification model by

TABLE 1: The cost matrix of binary classification.

Class attributes	True positive	True negative
Predict positive	$C(1, 1)$ or T_p	$C(1, 0)$ or F_p
Predict negative	$C(0, 1)$ or F_N	$C(0, 0)$ or T_N

changing the samples distribution. The latter two methods are emphatically introduced [40].

Assuming that the number of given the class labels of training set is C and the number of training samples in each category is N_i , the classification cost is defined as follows.

(1) $\text{Cost}[i, c]$ ($i, c \in \{1, \dots, C\}$) is described as the misclassification cost where the category I is misclassified as category c . Then, $\text{Cost}[i, i] = 0$ is obvious.

(2) $\text{Cost}[i]$ ($i \in \{1, \dots, C\}$) represents the total cost function of category i , namely, $\text{Cost}[i] = \sum_{c=1}^C \text{Cost}[i, c]$.

The cost expressions of oversampling, undersampling, and threshold adjusting by definition are discussed as follows.

In oversampling and undersampling, the cost expression of N_k^* is defined by

$$N_k^* = \left[\frac{\text{Cost}[k]}{\text{Cost}[\lambda]} N_\lambda \right], \quad (16)$$

where N_k^* is the number of categories k . λ represents the resample category of oversampling and undersampling, which is calculated by (17) and (18), respectively:

$$\lambda = \arg \min_j \frac{(\text{Cost}[j] / \min_c \text{Cost}[c]) N_{\arg \min_c \text{Cost}[c]}}{N_j}, \quad (17)$$

$$\lambda = \arg \max_j \frac{(\text{Cost}[j] / \max_c \text{Cost}[c]) N_{\arg \max_c \text{Cost}[c]}}{N_j}. \quad (18)$$

However, the realization principle of threshold adjusting can be interpreted as follows:

$$O_i^* = \eta \sum_{c=1}^C O_i \text{Cost}[i, c]. \quad (19)$$

In (19), O_i ($i \in \{1, \dots, C\}$) is the actual output of different output nodes of ELM, and it satisfies constraint condition $\sum_{i=1}^C O_i = 1$, $0 < O_i < 1$. η is the normalization coefficient of $\sum_{i=1}^C O_i^* = 1$. At the same time, the output of threshold adjusting also satisfies constraint condition $\sum_{i=1}^C O_i^* = 1$, $0 < O_i^* < 1$.

2.4. The Evaluation Indicators of Classification Model. The evaluation indicators of binary classification and multiclassification are introduced to validate the effectiveness of the proposed method in the section.

2.4.1. The Cost-Sensitive Evaluation Indicators of Binary Classification. In binary imbalanced learning, the cost matrix is shown in Table 1. It is generally recognized that the cost of correct classification is defined as $T_p = T_N = 0$.

Based on Table 1, the cost-sensitive evaluation indicators of binary classification are defined as follows.

```

(1) Input:  $x, y, \{K_p\}_{p=1}^m, q, C$ 
(2) Output:  $f_c(x), \alpha, \gamma$ 
(3) Initialization:  $\gamma = \gamma^0, t = 0, S^* = x, S_k (\forall x_k \in S_k \subset x)$ 
% The resample of the cost-sensitive dataset
(4) Calculate  $N_k^*$  and  $\lambda$  based on Eq. (16) and (17)
(5) Judgement
(6) If  $N_k^* > N_k$ ,
(7) The number of resample in the  $S_k$  is  $N_k^* - N_k$ 
(8) Updating  $S^*$ 
(9) Updating the sample  $x \leftarrow S^*$ 
(10) End
% Construct the multi-kernel function
(11) Repeat
(12) Calculate  $K(\cdot, \cdot; \lambda) = \sum_{p=1}^m \gamma_p^t K_p$ .
(13) Updating  $\alpha^t$  based on Eq. (13).
(14) Updating  $\gamma^{t+1}$  based on Eq. (14).
(15) Iteration  $t = t + 1$ ;
(16) Until  $\max\{|\gamma^{t+1} - \gamma^t|\} \leq 1e - 4$ 
(17) Calculate the optimal output function  $f_c(x)$  of the MKL-CS-ELM based on Eq. (15).

```

ALGORITHM 1: Oversampling example.

The classification accuracy of positive samples (AP):

$$AP = \frac{|T_P|}{|T_P| + |F_N|}. \quad (20)$$

The classification accuracy of negative samples (AN):

$$AN = \frac{|T_N|}{|T_N| + |F_P|}. \quad (21)$$

Global classification accuracy (Accuracy):

$$Accuracy = \frac{|T_P| + |T_N|}{|T_P| + |F_N| + |T_N| + |F_P|}. \quad (22)$$

In (20)~(22), $|\cdot|$ represents module operation.

2.4.2. The Cost-Sensitive Evaluation Indicator of Multiclassification. The cost-sensitive evaluation indicator of multiclassification is more complicated than binary classification. The indicator of robustness r_α referred to in [41] is introduced to describe the classification performance in multiclassification. The robustness indicator r_α is calculated by

$$r_\alpha = \frac{AV_{Cost_\alpha}}{\max_i AV_{Cost_i}}. \quad (23)$$

In (23), AV_{Cost_α} is average cost of method α . $\max_i AV_{Cost_i}$ represents the maximum average cost of the designed method. The indicator of robustness r_α is lower, and the robust performance of the method is better.

3. Classification Method of Imbalance Sample Distribution Based on MKL-CS-ELM

The main procedure of proposed MKL-CS-ELM method involves data preprocessing (data normalization and feature extraction), construction of multikernel function, and

cost-sensitive learning. The brief process of MKL-CS-ELM is shown in Figure 1. The detailed process of the proposed method is described in Algorithms 1 and 2. The oversampling process refers to Algorithm 1 and the principle of undersampling is similar to oversampling. Algorithm 2 is the implementation process of threshold adjusting method.

4. Experimental Description

4.1. The Principle of Check Valve and Experiment Platform

4.1.1. The Principle of Check Valve. The check valve completed a process of feeding and discharging in every stroke of the diaphragm pump. Assume that the stroke coefficient of diaphragm pump is 50 r/min and the reciprocating action of inlet and outlet check valve will be 72000 times when it is in the normal operation for one day. Therefore, the check valve is core component of frequent motion in diaphragm pump, and it also turns into one of the most important reasons for the check valve failure. The high pressure diaphragm pump and the failure check valve for mineral slurry pipe transportation with solid-liquid two-phase flow are shown in Figure 2.

In Figure 2, the check valve of the high pressure diaphragm pump is a cone-valve and its simple structure is shown in Figure 3. And “spool-spring” forms a weakly damped oscillation system. There are two reasons for the vibration of the system: one is external factor (resonance); the other is caused by its own characteristics. When the frequency of the external excitation source is an integral multiple of the natural frequency of the valve system, the resonance of the whole system will occur during work. So, the different running states of the check valve can be effectively judged by analyzing the vibration signal of the check valve.

4.1.2. Vibration Data Acquisition Experiment Platform. Figure 4 is the experiment platform of check valve. The

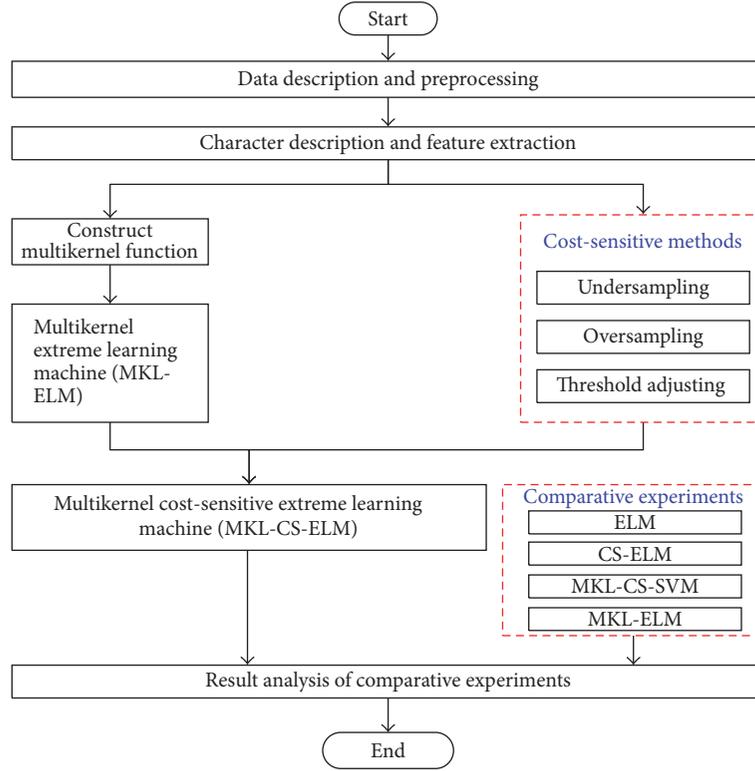


FIGURE 1: The realization process of MKL-CS-ELM.

- (1) Input: $x, \gamma, \{K_p\}_{p=1}^m, q, C$
- (2) Output: $f_c(x), \alpha, \gamma$
- (3) Initialization: $\gamma = \gamma^0, t = 0, S^* = x, S_k (\forall x_k \in S_k \subset x)$
% Construct the multi-kernel function
- (4) Repeat
- (5) Calculate $K(\cdot, \cdot; \lambda) = \sum_{p=1}^m \gamma_p^t K_p$.
- (6) Updating α^t based on Eq. (13).
- (7) Updating γ^{t+1} based on Eq. (14).
- (8) Iteration $t = t + 1$;
- (9) Until $\max\{\gamma^{t+1} - \gamma^t\} \leq 1e - 4$
- (10) Calculate the optimal output function $f_c(x)$ of the MKL-CS-ELM based on Eq. (15)
% Cost-sensitive Learning by Threshold Adjusting
- (11) Calculate the output O_i^* of MKL-CS-ELM based on the Eq. (19).
- (12) Calculate the optimal output $f_c(x) = \arg \max_i o_i^*$ of MKL-CS-ELM.

ALGORITHM 2: Threshold adjusting example.

three-cylinder diaphragm pump includes 3 pairs of check valves, which means that it includes 3 inlet check valves and 3 outlet check valves. So, in the process of data acquisition, the six PCB 352C33 accelerometers are installed on the check valve housing to collect vibration data by a PXI-3342. The data sampling frequency f_s is 2560 Hz and the data point N is 20480.

4.2. Experimental Setting. The data attributes of check valve and classification information are defined as in Table 2.

Based on the data characteristics in Table 2, the three kinds of cost matrixes are introduced and defined as follows [42].

(1)

$$\begin{aligned}
 & 1 < \text{Cost}[i, j] \leq 10 \quad j = c \cap j \neq i, \\
 & \text{Cost}[i, j \neq c] = 1 \quad j \neq i, \\
 & \text{Cost}[i] = \text{Cost}[i, c] \quad j \neq c, \\
 & \text{Cost}[c] = 1.
 \end{aligned} \tag{24}$$

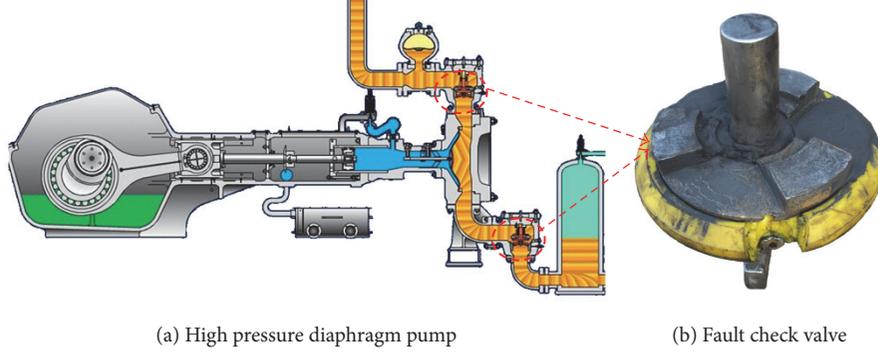


FIGURE 2: The high pressure diaphragm pump and fault check valve.

TABLE 2: The description of the experimental datasets.

Datasets	Data length	Attribute	Classes	Classification distribution
Check valve	20480	9	3	Normal condition (NC)/stuck valve fault (NK)/abrasion fault (NM)

(2)

$$\begin{aligned}
 1 \leq \text{Cost}[i, j] = H_i \leq 10 \quad j \neq i, \\
 \text{Cost}[i] = H_i, \\
 \exists H_i = 1.
 \end{aligned} \tag{25}$$

(3)

$$\begin{aligned}
 1 \leq \text{Cost}[i, j] \leq 10 \quad j \neq i, \\
 \text{Cost}[i] = \sum_{c=1}^C \text{Cost}[i, c], \\
 \exists \text{Cost}[i, j] = 1.
 \end{aligned} \tag{26}$$

4.3. The Feature Extraction of Wavelet Packet Energy Entropy.

Figure 5 shows the time and frequency waveform of the vibration signal for the check valve under 3 different operating conditions, including normal condition (NC), stuck valve fault (NK), and abrasion fault (NM). From point of the time domain and frequency domain waveform, it can be seen that the abnormal check valve has occurred, but further reasons or categories can not be obtained. In order to realize the automatic identification of the different running states of the check valve, it is necessary to extract the effective characteristics of the running state and then construct the state identification model.

The feature extraction makes full use of the advantage of wavelet packet and entropy in this paper. The third-layer wavelet packet energy distribution coefficient and energy entropy are extracted as characteristic parameters of the following classification model [42]. The selection of feature extraction method is based on the following points to consider.

(1) It is by using wavelet packet technique that the vibration signal of check valve can be mapped to wavelet-basis functions without information loss and has the superior ability in localization analysis of nonstationary signal.

(2) Entropy is introduced into depicting the operation state characteristics for check valve. This is mainly because the more disordered the system is, the greater the entropy becomes. And then, we can extract sensitive and transient features to describe the operation state of check valve.

The steps of feature extraction are listed below.

(1) Signal decomposition and reconstruction: the vibration signal of check valve is analyzed by three layers' wavelet packet transform to get the wavelet coefficients of the third-layer decomposition. In this paper, "db10 wavelet" is chosen as basic wavelet-basis function, which is mainly because "db10 wavelet" can well reflect the sensitive and transient features of vibration signal of check valve.

(2) Extraction feature vector: the wavelet packet energy distribution coefficient P_{3i} of reconstructed signals of the third-layer wavelet packets coefficients and energy entropy H compose the feature vector $T = [P_{30}, P_{31}, \dots, P_{37}, H]$. P_{3i} and H can be calculated as follows:

$$P_{3i} = \frac{\sqrt{E(a_{3i})}}{\sum_{i=1}^L \sqrt{E(a_{3i})}}, \tag{27}$$

$$H = - \sum_{i=1}^L p_{3i}^2 \log p_{3i}^2, \tag{28}$$

where L denotes the number of component signals ($L = 8$) and $E(a_{3i})$ represents the energy of the reconstruction signal of third-layer wavelet coefficients.

According to the definition of feature extraction in (27) and (28), the feature vectors of check valve can be calculated. Because of the limited space, partial features (not all features) are shown in Table 3. Compared to the normal check valve with the fault check valve, the operating conditions will be easily distinguished based on the wavelet packet energy distribution coefficient P and energy entropy H . It shows that the feature extraction method based on wavelet packet energy entropy is effective and reliable.

TABLE 3: The wavelet packet energy entropy features.

Fault class	Feature parameters								
	P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	H
NC	0.4521	0.2296	0.0693	0.0663	0.0365	0.0304	0.0779	0.0379	0.5848
NM	0.8257	0.1458	0.0066	0.0115	0.0001	0.0001	0.0067	0.0035	0.3453
NK	0.9263	0.0464	0.0132	0.0046	0.0001	0.0002	0.0060	0.0032	0.1469

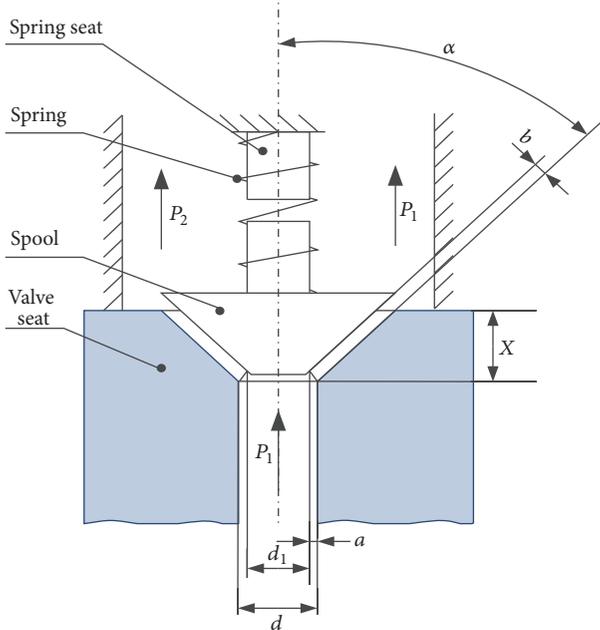


FIGURE 3: The structure diagram of cone-shaped check valve.

TABLE 4: The cost matrix of check valve.

Fault classes	NC	NK	NM
NC	0	1	1
NK	R	0	1
NM	R	1	0

5. Discussion of Experimental Results

Based on the definition of cost functions, the diagnosis cost matrix is constructed and shown in Table 4. The value of diagnostic cost R is from 1 to 5 ($R \in [1, 5]$) and increases by certain step length (usually 0.5) in the experiments.

In the experimental processing, the 110 data samples are collected, including 70 NC data samples, 20 NK data samples, and 20 NM data samples. The data samples of the check valve will be processed by combining the cost matrix shown in Table 4 with theoretical illustration of oversampling, undersampling, and threshold adjusting in Section 2.3. Then, the fault diagnosis classification models of ELM, CS-ELM, MKL-ELM, MKL-CS-ELM, and MKL-CS-SVM are constructed. The experimental results of binary classification and multiclassification for check valve are elaborated as follows in detail.

5.1. The Experimental Results Analysis of Binary Classification for Check Valve. In the binary classification experiments, the datasets of NC and NK are selected as the test data. The cost matrix is consistent with Table 4. The experimental results are described as follows.

5.1.1. The Experimental Results of Oversampling. The data sample distribution of oversampling is calculated and shown in Table 5 according to cost matrix in Table 4, (16), and (17). The 90 data samples are collected, 54 samples are selected as training samples, and the remaining 36 samples as test samples. Then the recognition results of classification models are presented in Figure 6.

As seen in Figure 6, some conclusions can be observed, including the following: (1) In the cost-sensitive processing of oversampling, the AP of CS-ELM, MKL-CS-SVM, and MKL-CS-SVM increases at first then decreases with increasing cost R , the AN increases at first then reaches steady state with increasing cost R , and the global classification accuracy (Accuracy) increases at first and then decreases with increasing cost R . (2) The recognition results of ELM and MKL-ELM method do not change with increasing cost R , which is mainly because the data distribution of the mentioned ELM and MKL-ELM does not also change. Therefore, it is only for the comparison of experimental results and independent of the diagnostic cost R . (3) In CS-ELM, MKL-CS-SVM, and MKL-CS-ELM method, the optimal recognition effect is obtained when the diagnostic cost is $R = 2.5$. (4) Compared with the ELM and MKL-ELM methods without diagnostic cost, the diagnostic cost can improve the accuracy and reliability of classification models in CS-ELM, MKL-CS-SVM, and MKL-CS-SVM method. (5) From the experimental results, we can also see that the multikernel learning mechanism is also helpful to further improve the diagnostic performance of the classification models. At the same time, Figure 6 also shows CS-ELM and MKL-CS-ELM are more sensitive to the cost R than the MKL-CS-SVM.

5.1.2. The Experimental Results of Undersampling. The data sample distribution of undersampling is calculated based on the Table 4, (16), and (18). Then, the recognition results of the above-mentioned classification models are displayed in Figure 7.

As shown in Figure 7, some conclusions can be obtained, which are similar to the results of oversampling methods. Moreover, the classification performance of mentioned classification models for check valve is slightly poor in undersampling. The experimental results found that the major

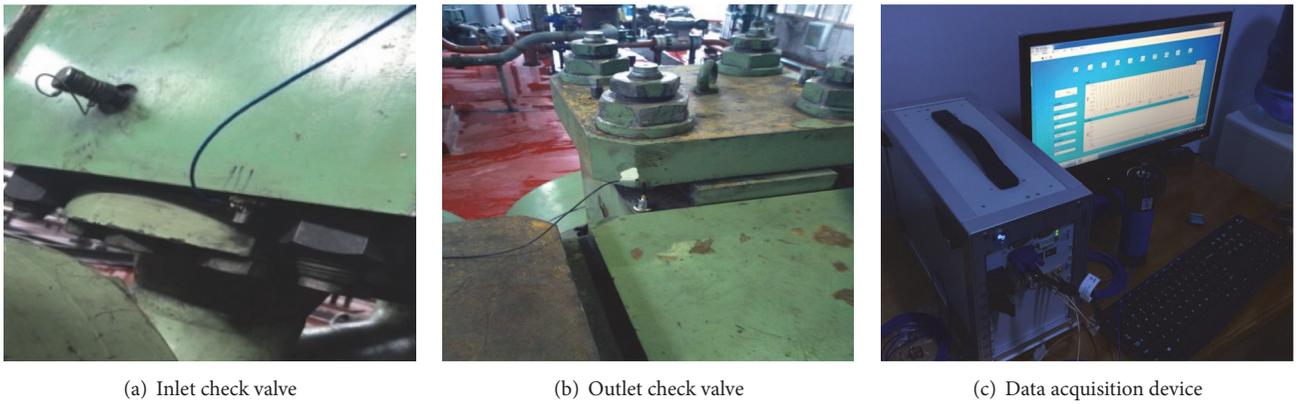


FIGURE 4: The experiment platform of check valve.

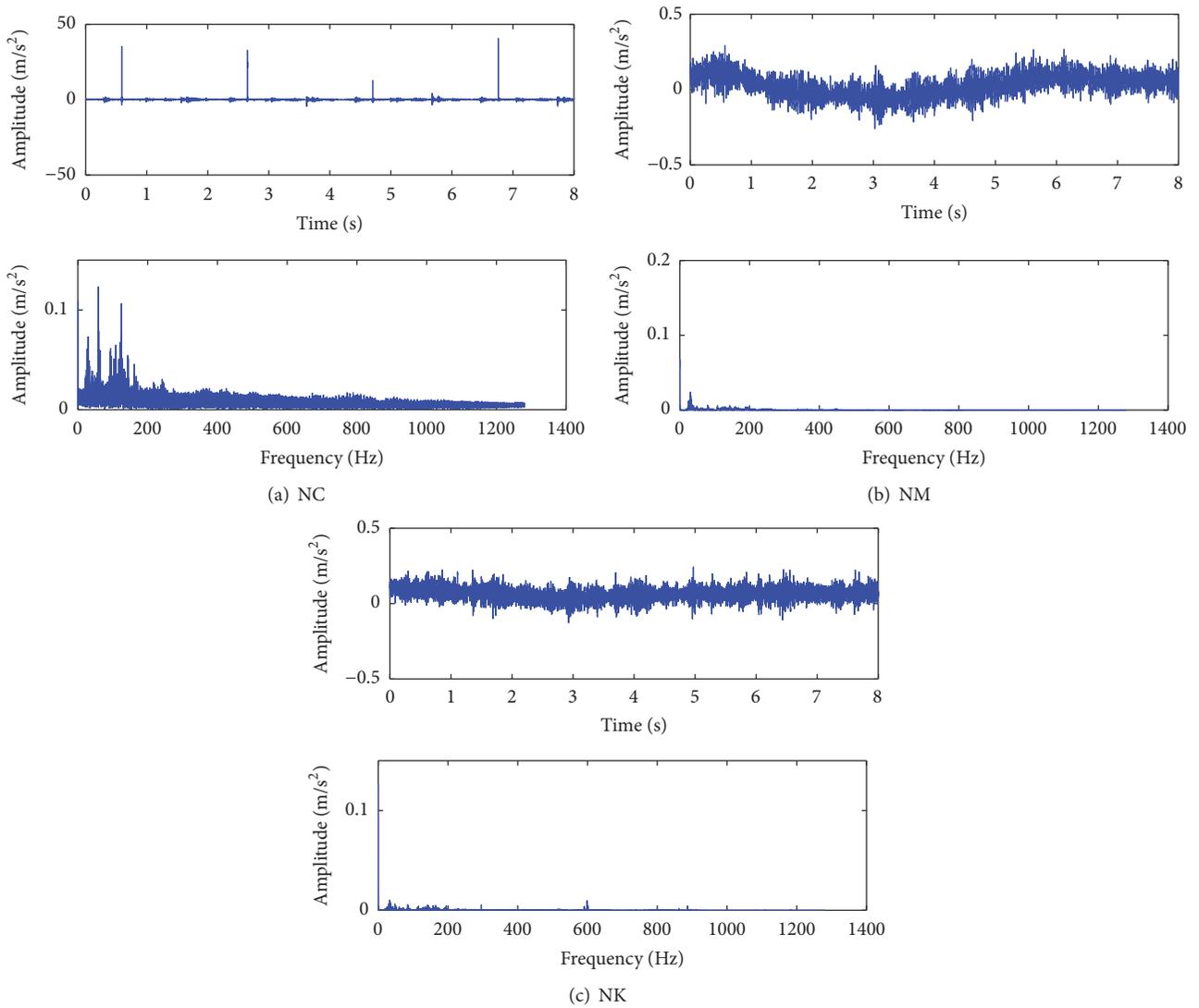


FIGURE 5: The analysis results of time domain and frequency domain for different work conditions.

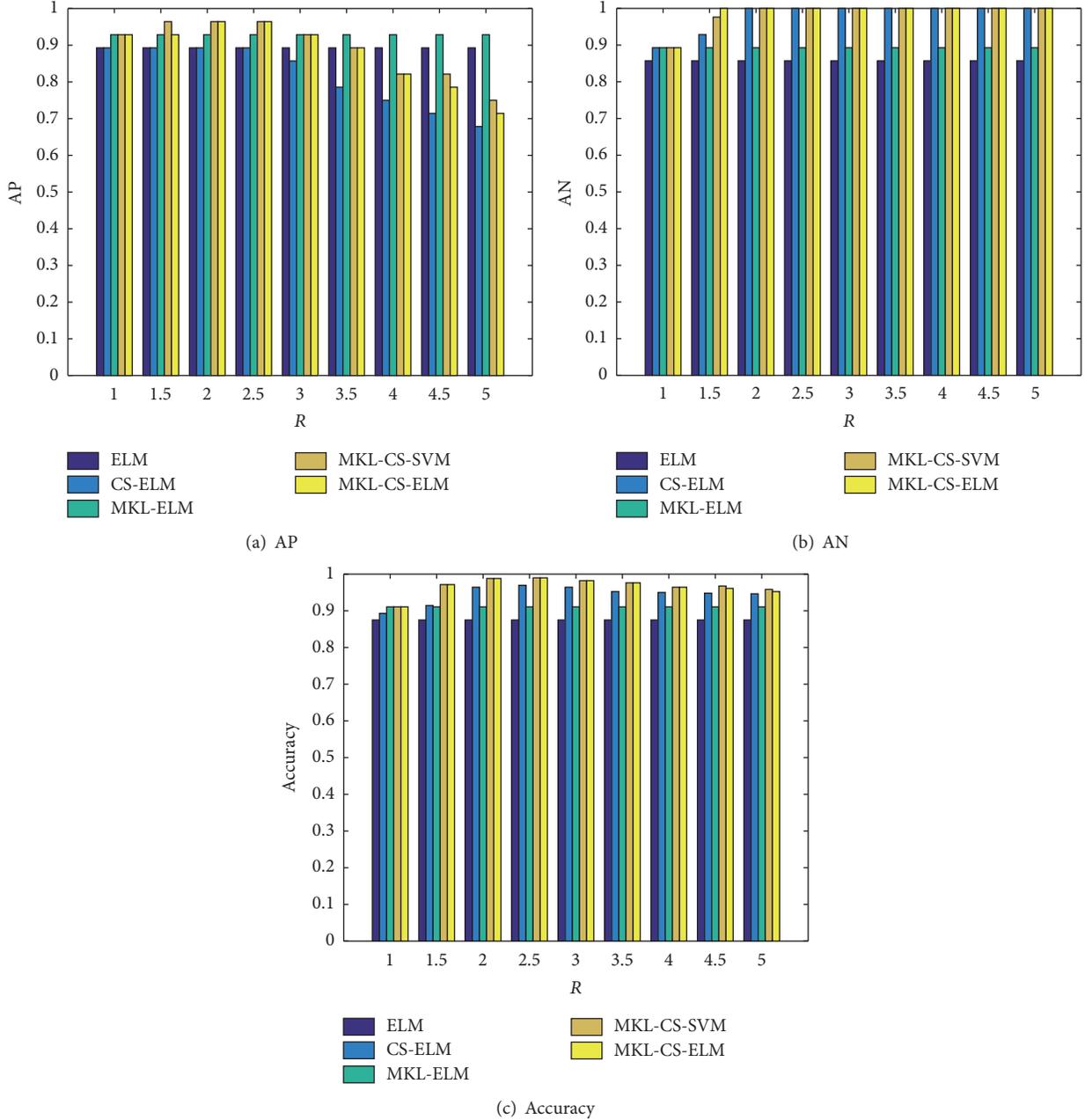


FIGURE 6: The recognition results of five classification models in oversampling.

problems are mostly owing to lack of the enough samples of check valve and the extreme imbalance of sample distribution is caused in the undersampling processing. At the same time, we can also observe an interesting phenomenon that when the sample is very small, the classification results of MKL-CS-ELM are slightly worse than the other classification models. This is probably an indirect argument that the training process of MKL-CS-ELM also needs the sufficient samples and the essence of MKL-CS-ELM is the single-hidden layer feedforward neural network. At the same time, the presented results also indirectly demonstrate the superiority of SVM in classification with smaller samples.

5.1.3. The Experimental Results of Threshold Adjusting. Based on the Table 4 and (19), the recognition results of five classification models in threshold adjusting are presented in Figure 8.

In Figure 8, the classification models of CS-ELM, MKL-CS-SVM, and MKL-CS-ELM can obtain good effects due to the introduction of cost-sensitive learning mechanism. The AN, AP, and Accuracy of aforementioned classification models are significantly improved with the increasing cost R . At the same time, the misclassification and missed diagnosis samples are sharply reduced with the increasing cost R . Compared with the performance of oversampling

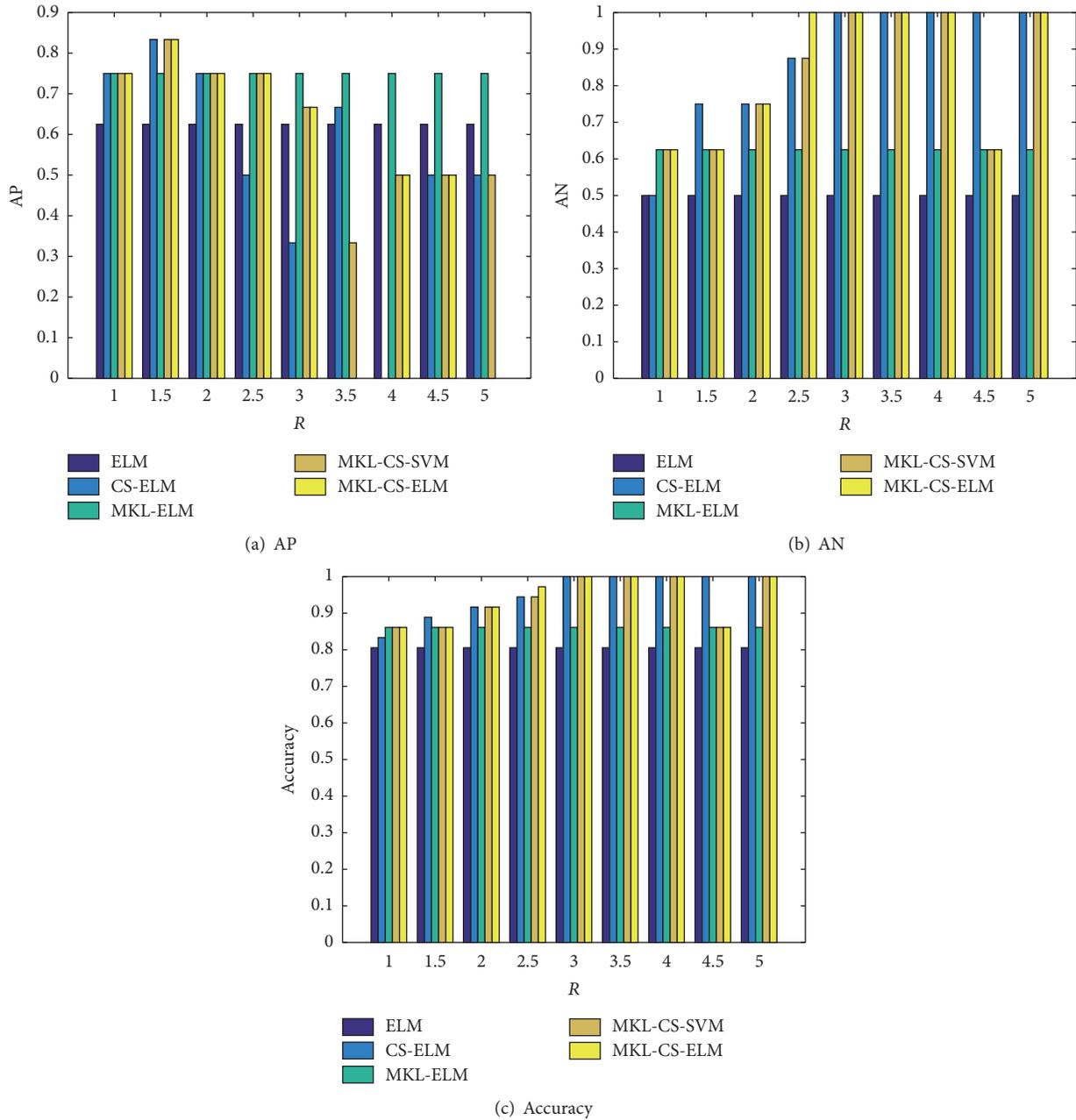


FIGURE 7: The recognition results of five classification models in undersampling.

and undersampling, the experimental results show that the threshold adjusting algorithm can also achieve satisfactory results. Therefore, the cost-sensitive method of threshold adjusting is also one of the effective choices for imbalance and inequality diagnosis cost in binary classification problems.

5.2. The Experimental Results Analysis of Multiclassification for Check Valve. In order to test validity and generalization ability of MKL-CS-ELM, the aforementioned three cost-sensitive methods are applied to identify multioperation states of check valve. Then the effectiveness of the proposed method is verified by multiclassification tests.

5.2.1. The Experimental Results of Oversampling. In the multiclassification experimental processing, the 110 data samples are collected, 66 samples are selected as training samples, and the remaining 44 samples are as test samples. The data sample distribution of oversampling is calculated based on (16) and (17). And the recognition results of classification models are presented in Figure 9.

As seen in Figure 9, the classification accuracy of CS-ELM, MKL-CS-SVM, and MKL-CS-ELM increases with the increasing cost R . On the contrary, the misclassification samples sharply reduce with the increasing cost R . The three classification models of CS-ELM, MKL-CS-SVM, and

TABLE 5: The data sample distribution by oversampling for check valve.

Cost R	Fault classes	Original samples	Processed samples
1	NC	70	70
	NK	20	70
1.5	NC	70	70
	NK	20	105
2	NC	70	70
	NK	20	140
2.5	NC	70	70
	NK	20	175
3	NC	70	70
	NK	20	210
3.5	NC	70	70
	NK	20	245
4	NC	70	70
	NK	20	280
4.5	NC	70	70
	NK	20	315
5	NC	70	70
	NK	20	350
--	--	--	--
--	--	--	--

MKL-CS-ELM can gain the optimal classification performance when the cost R is equal to 2.5 in oversampling processing. Meanwhile, compared with the experimental results illustrated in Figures 9(a), 9(b), and 9(c), some conclusions are summarized as follows: (1) The CS-ELM and MKL-CS-ELM are more sensitive to the cost R than the MKL-CS-SVM. (2) The classification performance of MKL-CS-ELM is slightly better than other above-mentioned classification models. (3) The change regularity of classification accuracy, misclassification, and missed diagnosis samples with the cost R is obtained and shown as follows: (1) the diagnosis cost $R = 2.5$ can be regarded as a demarcation line and inflection point of classification accuracy. (2) The misclassification and missed diagnosis samples drastically reduce when the cost R is less than 2.5. And the misclassification samples are reduced to 0 and reached balanced state when the cost R is greater than 2.5. But the missed diagnosis samples are sharply increased and the classification accuracy is also gradually decreasing. (3) The experimental results show that the above-mentioned cost-sensitive methods are feasible in check valve fault diagnosis of industrial field.

5.2.2. The Experimental Results of Undersampling. Similar to the previous oversampling approach, the data sample distribution of undersampling is calculated. Then, the experimental results of mentioned-above classification models are presented in Figure 10.

As shown in Figure 10, the classification accuracy of multikernel cost-sensitive diagnosis models is obviously decreased due to sharply reducing of data samples in undersampling. But, the misclassification samples can be also effectively restrained (even reduced to 0) by undersampling

when the cost R is equal to 2.5. However, Figure 10 also shows that the undersampling method should not be used in the conditions of the insufficient samples and high-accuracy requirements.

5.2.3. The Experimental Results of Threshold Adjusting. In the same way, the multiclassification recognition results of five mentioned classification models by the threshold adjusting are presented in Figure 11.

As depicted in Figure 11, in threshold adjusting processing, the misclassification samples are reduced to 0 when the cost R is increased to 2.5. The cost-sensitive classification models reach balanced state when the cost R is increased to 2.5, but the missed diagnosis samples and accuracy of CS-ELM, MKL-CS-SVM, and MKL-CS-ELM have no obvious change with the continuous increasing cost R .

5.3. Robust Performance Evaluation of Three Cost-Sensitive Methods for Check Valve. In order to assess the effectiveness of three cost-sensitive classification methods and choose the proper evaluation index for fault diagnosis of check valve, the robust performance evaluation r_α according to the description in Section 2.4.2 is calculated; the change regularity of robust performance index r_α varying with cost R is obtained and shown in Figure 12.

Figure 12 shows the comparative tests of robust performance evaluation in three cost-sensitive methods. The robust performance index r_α of the undersampling is biggest. That is to say, when the sample distribution is very imbalanced, it is not suitable to adopt the cost-sensitive method of undersampling. Moreover, in CS-ELM, MKL-CS-SVM, and MKL-CS-ELM method, the robust performance index r_α in

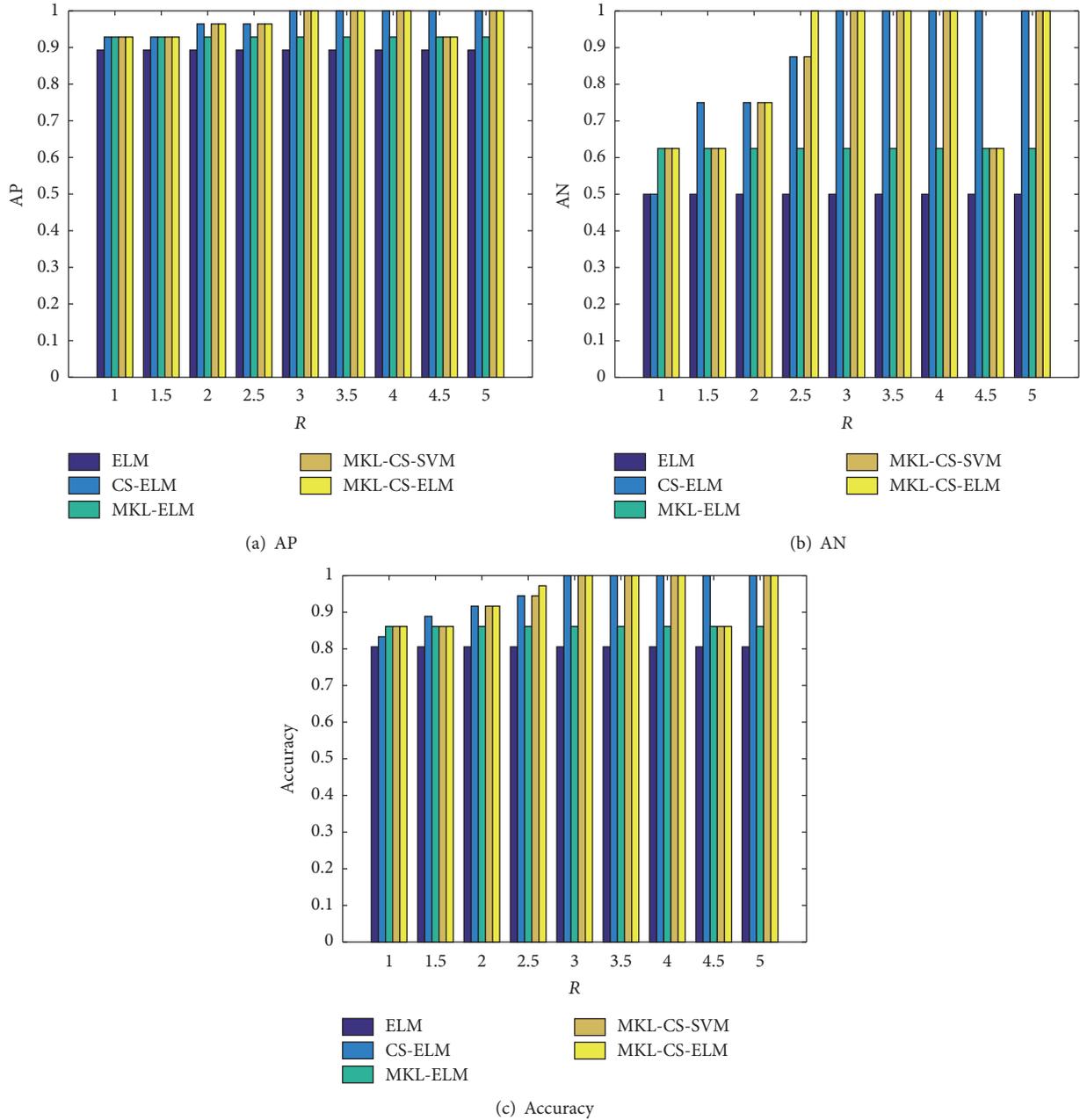


FIGURE 8: The recognition results of five classification models in threshold adjusting.

oversampling decreases at first then increases with increasing cost R and the robust performance index r_α in threshold adjusting decreases at first and then reaches steady state with increasing cost R . At the same time, Figure 12 also shows that the robust performance index r_α of oversampling is smaller than the threshold adjusting when the diagnosis cost R is less than 2.5, and then the change trend is reversed when the diagnostic cost R is greater than 2.5. Therefore, the oversampling and threshold adjusting are more appropriate cost-sensitive methods in multioperation states recognition of check valve.

6. Discussion and Conclusion

6.1. Discussion. High pressure diaphragm pump is often used as the core power equipment in slurry pipeline transportation, and its operating conditions are extremely complex. Therefore, it is critical to improve state recognition accuracy for ensuring operation safety and stability. However, the check valve is the core component of the high pressure diaphragm pump, and it is one of the most easily damaged and frequently replaced parts. Meanwhile, in the developed data acquisition system of check valve, the vibration data with

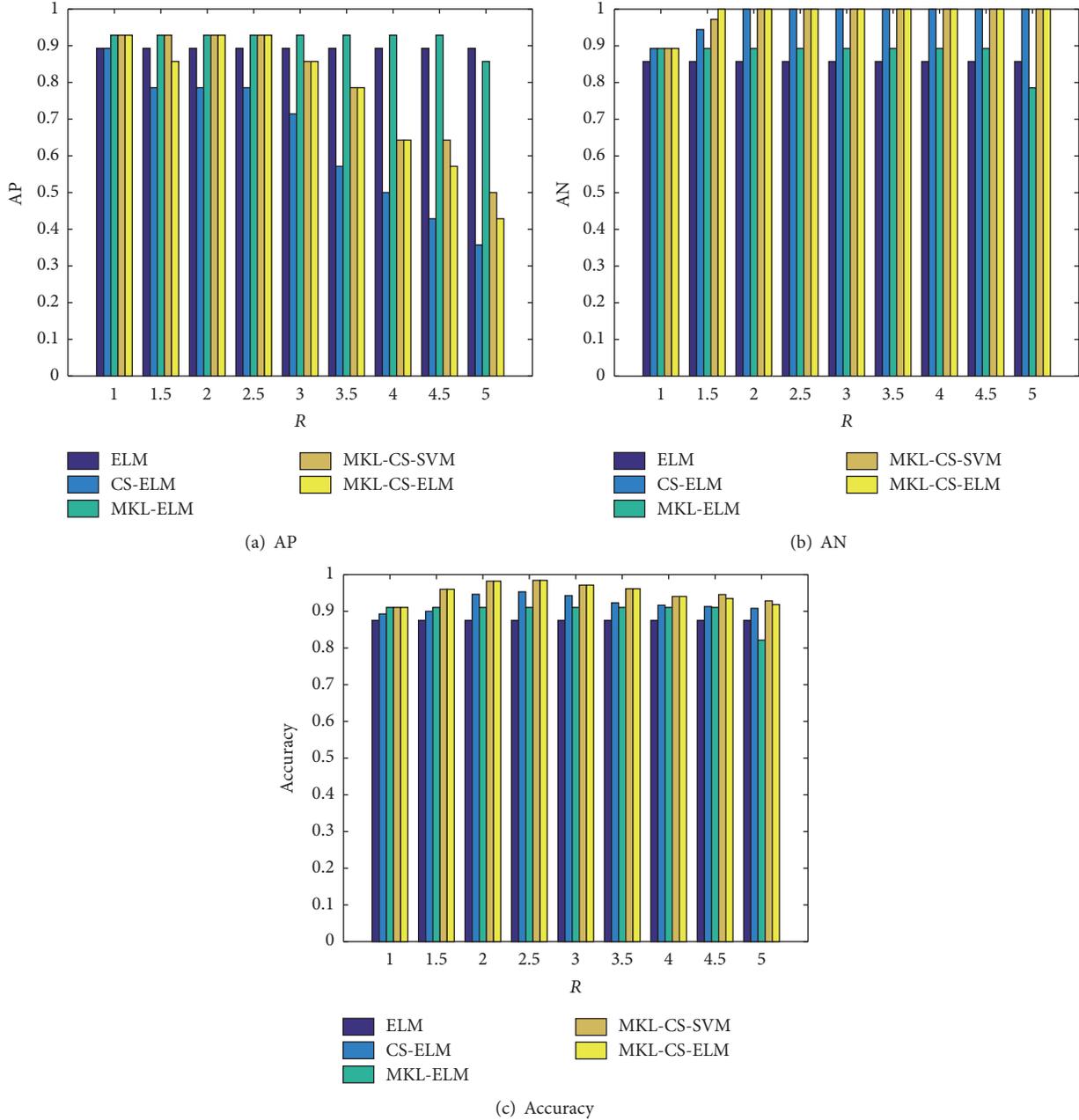


FIGURE 9: The recognition results of five classification models in oversampling.

normal operation has been collected in most of the time; on the contrary, the vibration data of fault time and fault state accounted for less. Therefore, it is of great significance to identify the operation state of the check valve effectively under the condition of complex operation and information asymmetry. Inspired by multikernel learning and cost-sensitive analysis, a fast diagnosis method of check valve based on MKL-CS-ELM is proposed. The presented MKL-CS-ELM method can complete the rapid positioning and analysis of the check valve fault and provide theoretical support for the adjustment and optimization in operation conditions of check valve during the follow-up operation.

(1) The multikernel learning mechanism is introduced to realize the multikernel projection of nonlinear and nonstationary data, which can overcome the limitation of incomplete information characterized with the single kernel function effectively and improve the ability to represent signals. Three kinds of common kernel function are used to construct multikernel classification model during the experiment. The introduction of multikernel learning can improve the recognition accuracy of classification model effectively through the analysis of MKL-ELM and ELM. In this case, what kind of kernel function and how many kernel functions are selected still lack normative choice mechanism. Therefore,

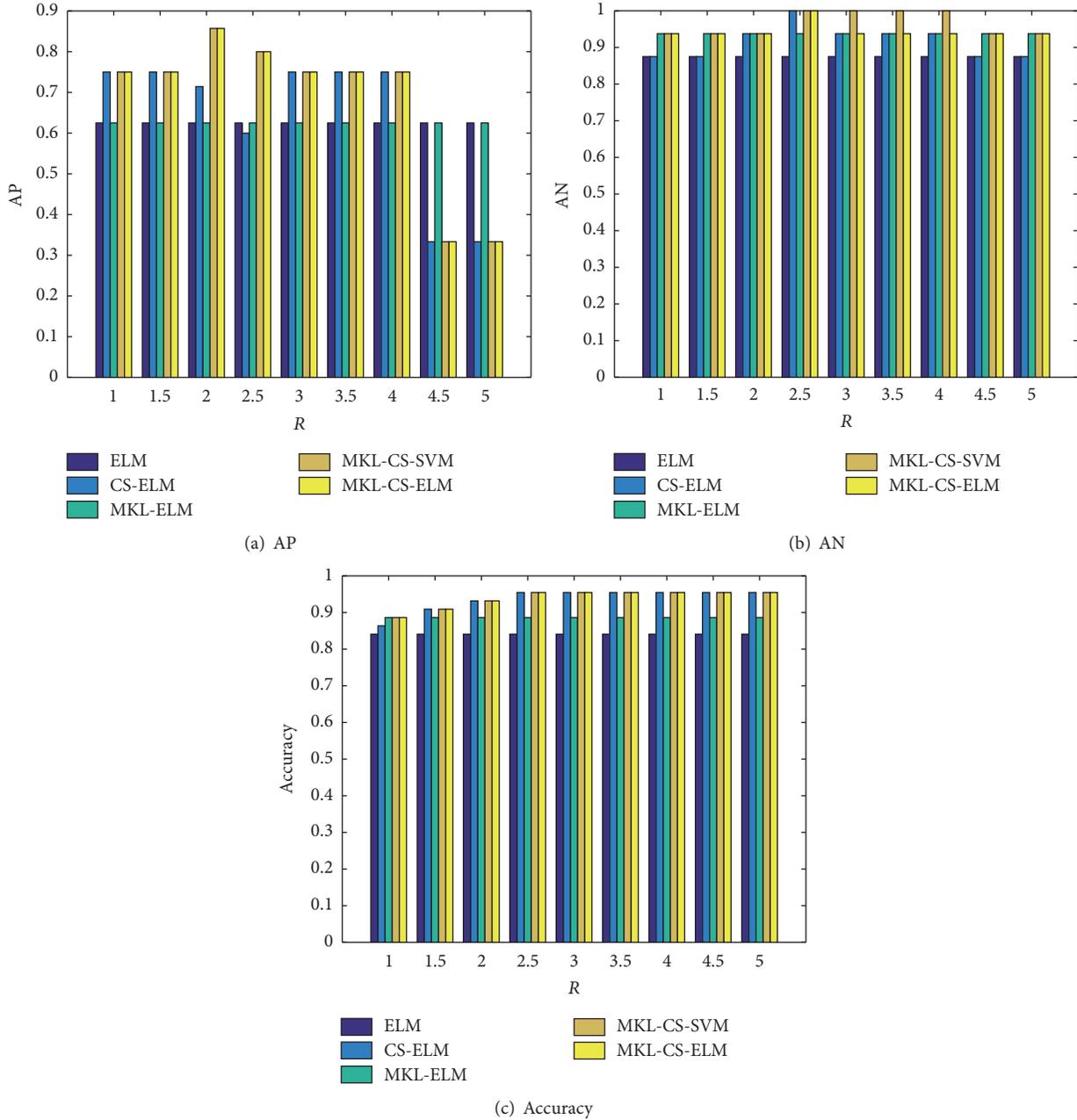


FIGURE 10: The recognition results of five classification models in undersampling.

we need to combine the signal characteristics and previous empirical rules about the selection to the kernel function so as to complete the selection of the effective kernel function and construct the multikernel function.

(2) In order to overcome the deficiency of assuming that the classification cost is equal through the classification model and improve the actual adaptability of the model, the paper makes the choice of the common cost-sensitive processing methods to construct CS-ELM model. The effectiveness of the introduction to cost-sensitive mechanism has been demonstrated through the binary classification and multiclassification recognition results; the experimental

results when using three kinds of cost-sensitive methods have also been compared with each other in different situations to provide theoretical support and guidance for the selection of cost-sensitive method. However, the cost of diagnosis needs to be moderate through the experimental comparison; otherwise it will reduce the overall recognition accuracy of the classification model.

6.2. Conclusion. The fault diagnosis model of MKL-CS-ELM based on the multikernel learning and cost-sensitive learning is constructed, and the datasets of check valve are used to verify the effectiveness of the proposed method. By

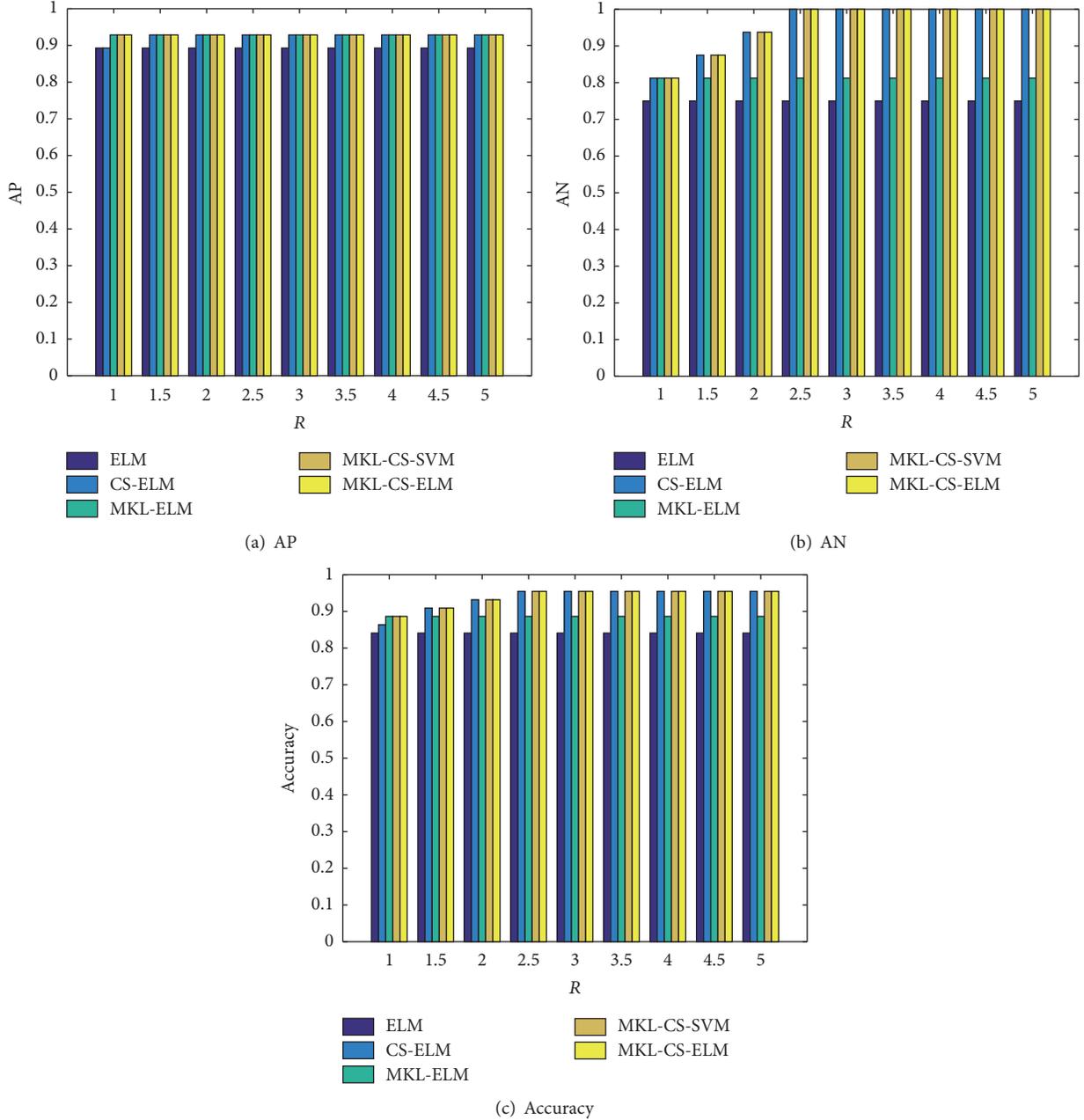


FIGURE 11: The recognition results of five classification models in threshold adjusting.

comparative tests, some conclusions can be summarized as follows.

(1) The MKL-CS-ELM can gain fair or better performance than the other classification models, including ELM, CS-ELM, MKL-ELM, and MKL-CS-SVM.

(2) The comparative analysis of robust performance evaluation r_α demonstrates that the oversampling and threshold adjusting cost-sensitive method are more appropriate choice in multiclassification application of check valve.

(3) The study of three cost-sensitive methods shows that, by selecting the appropriate cost R , the constructed

classification model can reduce the misclassification rate, achieve the balance between misclassification rates, miss diagnosis rate, and accuracy, and also improve the overall reliability of the classification model.

(4) The overall experimental results of the check valve show that the theory of multikernel learning and cost-sensitive learning can effectively overcome the disadvantage of the sample distribution imbalance and diagnostic cost equalization supposed in the conventional classification model and improve the accuracy and reliability of classification models.

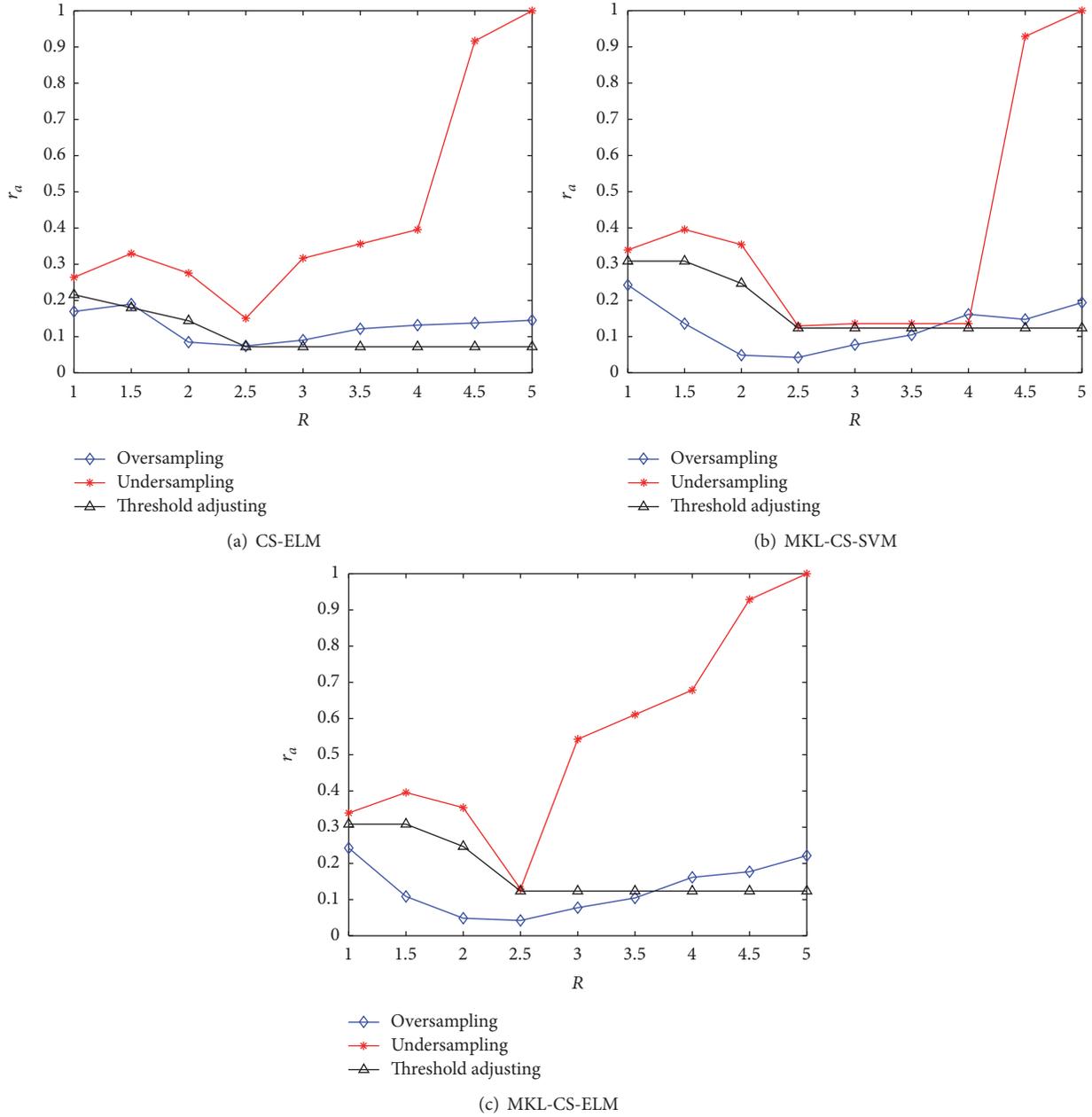


FIGURE 12: The performance comparisons of robust performance evaluation for different cost-sensitive methods.

Abbreviations

ELM: Extreme learning machine
 MKL-ELM: Multikernel ELM
 MKL-CS-ELM: Multikernel cost-sensitive ELM
 RBF: Radial basis function
 KKT: Karush Kuhn Tucker
 NK: Stuck valve fault
 C: Regularization parameter
 AP: The classification accuracy of positive samples
 $\{\kappa_p(\cdot, \cdot)_{p=1}^m\}$: The number of basic kernel functions is m

$\kappa(\cdot, \cdot; \gamma)$: The typical form of multikernel function
 $\{\phi(\cdot)\}_{p=1}^m$: The high dimensional feature mapping of $\{\kappa_p(\cdot, \cdot)_{p=1}^m\}$
 SVM: Support vector machine
 CS-ELM: Cost-sensitive ELM
 MKL-CS-SVM: Multikernel cost-sensitive SVM
 LSSVM: Least squares SVM
 NC: Normal condition
 NM: Abrasion fault
 Accuracy: Global classification accuracy
 AN: The classification accuracy of negative samples

- γ_p : The combination coefficients of basic kernel functions
- $\phi(\cdot, \gamma)$: The high dimensional feature mapping of $\kappa(\cdot, \cdot; \gamma)$.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (51765022, 61663017, and 51169007) and Science & Research Program of Yunnan Province (2015ZC005).

References

- [1] E. Rogatsky, M. Gallitto, and D. T. Stein, "A novel performance test for outlet check valve function in HPLC pumps," *Life Science Instruments*, vol. 28, no. 6, pp. 30-31, 2010.
- [2] B. Yang and J. Zhang, "The automatic diagnosis technology of the one-way valve malfunction of the reciprocating piston diaphragm pump," *Technology Information*, vol. 7, pp. 116-117, 2012.
- [3] B. Liu and S.-F. Ling, "On the selection of informative wavelets for machinery diagnosis," *Mechanical Systems and Signal Processing*, vol. 13, no. 1, pp. 145-162, 1999.
- [4] W. Li, F. Gu, A. D. Ball, A. Y. T. Leung, and C. E. Phipps, "A study of the noise from diesel engines using the independent component analysis," *Mechanical Systems and Signal Processing*, vol. 15, no. 6, pp. 1165-1184, 2001.
- [5] N. Lawrence and H. Y. P. Kortekaas, "DECSIM—a PC-based diesel engine cycle and cooling system simulation program," *Mathematical and Computer Modelling*, vol. 33, no. 6-7, pp. 565-575, 2001.
- [6] R. A. Ogle and D. T. Morrison, "Investigation of an acid spill caused by the failure of an air-operated diaphragm pump," *Process Safety Progress*, vol. 20, no. 1, pp. 41-49, 2001.
- [7] W. Liu and T. Chen, "Research on condition monitoring and trend prediction of reciprocating pump based on grey-neural network," *Journal of Safety Science & Technology*, vol. 3, no. 1, pp. 79-84, 2013.
- [8] C. Wang, M. Gan, and C. Zhu, "Non-negative EMD manifold for feature extraction in machinery fault diagnosis," *Measurement*, vol. 70, pp. 188-202, 2015.
- [9] S. Lu, J. Wang, and Y. Xue, "Study on multi-fractal fault diagnosis based on EMD fusion in hydraulic engineering," *Applied Thermal Engineering*, vol. 103, pp. 798-806, 2016.
- [10] S. Guo, Y. C. Xu, X. S. Li, R. Tao, K. Li, and M. Gou, "Research on roller bearing with fault diagnosis method based on EMD and BP neural network," *Advanced Materials Research*, vol. 1014, no. 1014, pp. 501-504, 2014.
- [11] H. Li, Y. Hu, F. Li, and G. Meng, "Succinct and fast empirical mode decomposition," *Mechanical Systems and Signal Processing*, vol. 85, pp. 879-895, 2017.
- [12] S.-W. Fei, "Fault diagnosis of bearing based on wavelet packet transform-phase space reconstruction-singular value decomposition and SVM classifier," *Arabian Journal for Science and Engineering*, vol. 42, no. 5, pp. 1967-1975, 2017.
- [13] S. Guo, S. McLaughlin, X. Yongcheng, and P. White, "Theoretical and experimental analysis of bispectrum of vibration signals for fault diagnosis of gears," *Mechanical Systems and Signal Processing*, vol. 43, no. 1-2, pp. 76-89, 2014.
- [14] Z. Zheng, W. Jiang, Z. Wang, Y. Zhu, and K. Yang, "Gear fault diagnosis method based on local mean decomposition and generalized morphological fractal dimensions," *Mechanism and Machine Theory*, vol. 91, article no. 2479, pp. 151-167, 2015.
- [15] Z. Meng and L.-L. Li, "Rolling bearing fault diagnosis based on local characteristic-scale decomposition and morphological fractal dimension," *Acta Metrologica Sinica*, vol. 37, no. 3, pp. 284-288, 2016.
- [16] J. K. Lee, T. Y. Kim, H. S. Kim, J.-B. Chai, and J. W. Lee, "Estimation of probability density functions of damage parameter for valve leakage detection in reciprocating pump used in nuclear power plants," *Nuclear Engineering and Technology*, vol. 48, no. 5, pp. 1280-1290, 2016.
- [17] D. P. Lewis, T. Jebara, and W. S. Noble, "Nonstationary kernel combination," in *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pp. 553-560, ACM, Pittsburgh, Pa, USA, June 2006.
- [18] S. O. Cheng, A. J. Smola, and R. C. Williamson, "Learning the kernel with hyperkernels," *Journal of Machine Learning Research (JMLR)*, vol. 6, no. 1, pp. 1043-1071, 2005.
- [19] J. He, S.-F. Chang, and L. Xie, "Fast kernel learning for spatial pyramid matching," in *Proceedings of the 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pp. 1-7, June 2008.
- [20] M.-Z. Tang, C.-H. Yang, W.-H. Gui, and Y.-F. Xie, "Cost-sensitive probabilistic neural network with its application in fault diagnosis," *Control and Decision*, vol. 25, no. 7, pp. 1074-1078, 2010.
- [21] Y. Wang, F. Cao, and Y. Yuan, "A study on effectiveness of extreme learning machine," *Neurocomputing*, vol. 74, no. 16, pp. 2483-2490, 2011.
- [22] C. Yang, K. Huang, H. Cheng, and Y. Li, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems Man Cybernetics Systems*, vol. 47, no. 8, pp. 2398-2409, 2017.
- [23] R. Zhang, Y. Lan, G.-B. Huang, and Z.-B. Xu, "Universal approximation of extreme learning machine with adaptive growth of hidden nodes," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 2, pp. 365-371, 2012.
- [24] A. Lendasse, Q. He, Y. Miche, and G.-B. Huang, "Advances in extreme learning machines (ELM2012)," *Neurocomputing*, vol. 128, no. 6, pp. 1-3, 2014.
- [25] Q. Yu, Y. Miche, E. Eirola, M. van Heeswijk, E. Séverin, and A. Lendasse, "Regularized extreme learning machine for regression with missing data," *Neurocomputing*, vol. 102, no. 2, pp. 45-51, 2013.
- [26] T. Wang, D. Zhao, and Y. Feng, "Two-stage multiple kernel learning with multiclass kernel polarization," *Knowledge-Based Systems*, vol. 48, no. 2, pp. 10-16, 2013.
- [27] Z. Liang, S. Xia, Y. Zhou, and L. Zhang, "Training L_p norm multiple kernel learning in the primal," *Neural Networks*, vol. 46, no. 5, pp. 172-182, 2013.
- [28] X. Chao and Y. Peng, "A Cost-sensitive Multi-criteria quadratic programming model," *Procedia Computer Science*, vol. 55, pp. 1302-1307, 2015.
- [29] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2125-2136, 2017.

- [30] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. pp, no. 99, pp. 1–12, 2017.
- [31] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.
- [32] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [33] X. Liu, L. Wang, G. B. Huang, J. Zhang, and J. Yin, "Multiple kernel extreme learning machine," *Neurocomputing*, vol. 149, part A, pp. 253–264, 2015.
- [34] E. Zheng, C. Zhang, X. Liu, H. Lu, and J. Sun, "Cost-sensitive extreme learning machine," in *International Conference on Advanced Data Mining and Applications*, pp. 478–488, Springer, Berlin, Germany, 2013.
- [35] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.
- [36] G. Huang, G.-B. Huang, S. Song, and K. You, "Trends in extreme learning machines: a review," *Neural Networks*, vol. 61, pp. 32–48, 2015.
- [37] Z. Xu, R. Jin, H. Yang, and M. R. Lyu, "Simple and efficient multiple kernel learning by group Lasso," in *International Conference on Machine Learning*, pp. 1175–1182, 2010.
- [38] H. Yang, Z. Xu, J. Ye, I. King, and M. R. Lyu, "Efficient sparse generalized multiple kernel learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 22, no. 3, pp. 433–446, 2011.
- [39] W. Wu and J. Hu, "Fault diagnosis based on cost-sensitive transduction inference," *Chinese Journal of Scientific Instrument*, vol. 31, no. 5, pp. 1023–1028, 2010.
- [40] Z. H. Zhou and X. Y. Liu, "Training cost-sensitive neural networks with methods addressing the class imbalance problem," *IEEE Transactions on Knowledge and Data Engineering*, vol. 18, no. 1, pp. 63–77, 2006.
- [41] K. M. Ting, "An instance-weighting method to induce cost-sensitive trees," *IEEE Transactions on Knowledge and Data Engineering*, vol. 14, no. 3, pp. 659–665, 2002.
- [42] J. Wang, Q. Liu, and D. Qinhu, "Nonlinear membership function established by single-shot clustering method," *Journal of Zhengzhou University (Engineering Science)*, vol. 33, no. 2, pp. 28–30, 2012.

Research Article

RBF Nonsmooth Control Method for Vibration of Building Structure with Actuator Failure

Jianhui Wang,¹ Chunliang Zhang,¹ Houyao Zhu,¹ Xiaofang Huang,² and Li Zhang³

¹School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou 510006, China

²Engineering Earthquake Resistance Center, Guangzhou University, Guangzhou 51045, China

³Guangzhou Real Estate Management Vocational School, Guangzhou 510320, China

Correspondence should be addressed to Chunliang Zhang; nhzcl@163.com

Received 13 July 2017; Accepted 30 October 2017; Published 20 December 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Jianhui Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In order to accommodate the actuator failure, the finite-time stable nonsmooth control method with RBF neural network is used to suppress the structural vibration. The traditional designed control methods neglect influence of actuator failure in structural vibration. By Lyapunov stable theory, the designed control method is demonstrated to suppress the building structural vibration with actuator failure. Finally, there are some examples to numerically simulate the three-layer building structure which is affected by El Centro seismic wave. Control effect of nonsmooth control is compared with no control and LQR control. The simulation results demonstrate that the designed control method is great for vibration of building structure with actuator failure and great antiseismic effect.

1. Introduction

How to reduce the severe and persistent vibration for structure under the earthquake vibration and wind resistance is a hot topic. In the last few decades, the structural vibration control was proved as an active and effective measure to suppress vibration. And international and domestic academics have brought up many kinds of effective control algorithm such as LQR (Linear Quadratic Regulator), LQG (Linear Quadratic Gaussian), ILC (Iterative Learning Control), and pole placement [1–5]. The aforementioned results do not consider the case of actuator failure. However, actuator failure is inevitable in the real project.

Thus, more and more researchers make contributions to control strategy in the case of actuator failure. And many effective ways have some development in respect of compensation of actuator failure. The study of actuator failure is started with dealing with linear system fault [6, 7]. Nevertheless, the control of actuator failure is very limited in the application of building structure.

In addition, the convergence of control system is an important index [8]. However, many linear control methods are to make system Lyapunov stable. What is more, they

belong to asymptotic stable research field that motion track is converged to the system's equilibrium point in the case of time that tends to infinity. In the view of making control system of structural vibration rapidly stabilize, it is necessary to study control methods making closed-loop system converge in finite time.

With the study and development of Lyapunov stable theory [9] and theorem of homogeneity, continuous nonsmooth control has made certain breakthrough [8–10]. Nonsmooth control has been widely applied [11–15] such as attitude control of spacecraft [12], high-precision guidance laws [14], and position control of permanent magnet synchronous motors [15]. Nevertheless, this control method is not applied on the building structure. At the same time, it cannot approximate well for uncertain part. The problem causes difficulty and challenge for design and analysis of control method. By learning literature [16–25], it is not hard to find out that neural network has wide prospect. And the neural network has great approximation effect for unknown model. Meanwhile, RBF neural network has great generalization and approximates any nonlinear function at random.

The paper carries out mathematical modeling and analysis for a building structure. According to the RBF neural

network, the seismic wave is made by autoadaptable approximation. Then, according to finite-time stable theory and analysis of actuator failure, the finite-time stable nonsmooth algorithm is designed for the problem of structural vibration. Finally, the control system is under seismic wave called El Centro. And numerical analysis of the strong nonlinear model is studied. The control effects of nonsmooth control and LQR control are analyzed contrastively.

The main contributions of this dissertation are as follows: The impact of uncertain actuator failures on building structure vibration is considered. Meanwhile, the actuator failure is compensated with RBF neural network. Building structure vibration is suppressed in a fast speed by applying the method of finite-time nonsmooth vibration control, which prevents building structure from vibration in a long time.

2. The Modeling and Analysis of Building Structure

Interlaminar shear model is used. The n layers' building structure is simplified into building structure n degrees of freedom. Effected under one-dimensional horizontal earthquake, the equation of motion is as follows [1]:

$$M\ddot{D} + C\dot{D} + KD = F\ddot{x}_g + EU. \quad (1)$$

In this equation, $D = [d_1, d_2, \dots, d_n]^T$ is displacement vector of the structure relative to the ground, where d_i ($i = 1, 2, \dots, n$) is displacement of the building structural i th floor relative to the ground. M is mass matrix. C is damping matrix. K is stiffness matrix. $F = MI$ is transform matrix of the ground seismic acceleration where $I = [1 \ 1 \ \dots \ 1]^T$ is the unit column of $n \times 1$. \ddot{x}_g is the ground seismic acceleration. E is a matrix denoting the location of actuators. U is the control input.

We define a state-space vector $S = [S_1, S_2]^T$, where $S_1 = [s_{11}, s_{12}, \dots, s_{1n}]^T = D$ and $S_2 = [s_{21}, s_{22}, \dots, s_{2n}]^T = \dot{D}$. Space state equations of (1) can be formulated as [26]

$$\dot{S} = A_r S + W_r \ddot{x}_g + B_r U, \quad (2)$$

where

$$\begin{aligned} A_r &= \begin{bmatrix} 0 & I_{n \times n} \\ -M^{-1}K & -M^{-1}C \end{bmatrix}, \\ W_r &= \begin{bmatrix} 0 \\ M^{-1}F \end{bmatrix}, \\ B_r &= \begin{bmatrix} 0 \\ M^{-1}E \end{bmatrix}. \end{aligned} \quad (3)$$

According to the rank criterion, the system (see (2)) is controllable. Hence, structural vibration can be suppressed effectively via designing control variable.

According to the finite-time stability theory, considering the motion equation of the structure, an actuator has been installed in each layer. E is $n \times n$ full rank matrix called

invertible matrix. We use variable $V = [u_1, u_2, \dots, u_n]^T$ and choose

$$U = E^{-1} (C\dot{D} + KD - F\ddot{x}_g + MV). \quad (4)$$

Equation (4) is plugged into (2) as

$$\dot{S} = \begin{bmatrix} 0 & I_{n \times n} \\ 0 & 0 \end{bmatrix} S + \begin{bmatrix} 0 \\ I_{n \times n} \end{bmatrix} V. \quad (5)$$

The system can be decomposed into n mutual independent subsystems as

$$\begin{aligned} \dot{s}_{11} &= s_{21}, \\ \dot{s}_{21} &= u_1, \\ \dot{s}_{12} &= s_{22}, \\ \dot{s}_{22} &= u_2, \\ &\vdots \\ \dot{s}_{1n} &= s_{2n}, \\ \dot{s}_{2n} &= u_n. \end{aligned} \quad (6)$$

The i th actuator failure mathematic model can be modeled as

$$\begin{aligned} u_i &= \rho_i v_i + u_{ki}, \quad \forall t \geq t_{iF}, \\ \rho_i u_{ki} &= 0, \end{aligned} \quad (7)$$

where $0 \leq \rho_i \leq 1$ and u_{ki} and t_{iF} are uncertain constants. When the constants $\rho_i = 1$ and $u_i = v_i$, this indicates that the i th actuator works normally (i.e., the actuators work in the failure-free case). Thus, the following 2 patterns of failures are considered.

- (1) $0 \leq \rho_i \leq 1$: this case indicates that the systems lose partial performance during the operation, which is known as Partial Loss of Effectiveness (PLOE); that is, $u_i = \rho_i v_i$.
- (2) $\rho_i = 0$: this case implies that actuator output u_i is no longer affected by v_i . v_i indicates Total Loss of Effectiveness (TLOE); that is, $u_i = u_{ki}$.

According to the above analysis, system mathematical model is rewritten as follows:

$$\begin{aligned} \dot{s}_{1i} &= s_{2i}, \\ \dot{s}_{2i} &= u_i, \\ \forall t &\geq t_{iF}, \quad i = 1, 2, \dots, n. \end{aligned} \quad (8)$$

3. Design of Control Algorithm

In the failure of the period, the controller is designed as $v_i = k^T w$, $w = (\alpha, 1)$, and $k = (k_{1i}, k_{2i})$ for any i th subsystem,

where α is designed as the finite-time stable nonsmooth control law [27].

$$\alpha = -k_1 \cdot \text{sign}(s_{1i}) \cdot |s_{1i}|^{\alpha_1} - k_2 \cdot \text{sign}(s_{2i}) \cdot |s_{2i}|^{\alpha_2}, \quad (9)$$

where $k_1 > 0$, $k_2 > 0$, $0 < \alpha_1 < 1$, $\alpha_2 = 2\alpha_1/(1 + \alpha_1)$, and $i = 1, 2, \dots, n$.

On the basis of the above controller design, in order to guarantee the system stability, we must design k to meet the following formula:

$$\rho_i k^T w = \alpha - u_{ki}. \quad (10)$$

However, the failure model parameters of the actuator are unknown so that \hat{k} is defined as the estimated value for k and $\tilde{k} = k - \hat{k}$ is defined.

The above analysis includes unknown seismic wave \ddot{x}_g disturbance, so subsequent analysis faces difficulty and challenge. Thus, the paper uses RBF neural network to approximate \ddot{x}_g .

RBF network has characteristics of universal approximation. We use theory that uses RBF network to approximate $f(x)$. The network algorithm is as follows:

$$h_j = \exp\left(-\frac{\|x - c_j\|^2}{2b_j^2}\right), \quad j = 1, 2, \dots, N, \quad (11)$$

$$f = W^{*T} h(x) + \varepsilon,$$

where x is an input of network, j is the j th joint of network's hidden layer, N is the number of network's hidden layers, $h = [h_1, h_2, \dots, h_N]^T$, W^* is the desirable permission of network, ε is an approximation error of network, and $\varepsilon \leq \varepsilon_N$.

The input of network is $x = [x_1 \ x_2]^T$. Then the output of network is as follows:

$$\hat{f}(x) = \widehat{W}^T h(x). \quad (12)$$

According to RBF theory, we make the following definitions: $s = [D \ \dot{D}]$, $\hat{f}(s) = \widehat{W}h(s)$, and $f(s) = \ddot{x}_g = W^*h(s)$. It is proved accordingly that the nonsmooth control law can make system globally finite-time stable. We can prove the following: the i th subsystem of Lyapunov function is built as

$$V_i = \frac{\rho_i}{2} \tilde{k}^T \Gamma^{-1} \tilde{k} + \frac{k_1}{1 + \alpha_1} |s_{1i}|^{1 + \alpha_1} + \frac{1}{2} s_{2i}^2 + \frac{1}{2\lambda} \widehat{W}^T \tilde{W}, \quad (13)$$

where Γ is definite matrix. $\lambda > 0$; $\widehat{W} = W^* - \tilde{W}$.

Setting $\dot{\tilde{k}} = -s_{2i} \Gamma^{-1} w$ and $\dot{\tilde{W}} = \lambda h(s)$, then

$$\dot{V}_i = k_1 |s_{1i}|^{\alpha_1} |\dot{s}_{1i}| + \alpha s_{2i} = -k_2 |s_{2i}|^{1 + \alpha_2}. \quad (14)$$

Obviously, \dot{V}_i is half negative. Therefore, the system is stable. According to the invariance principle, subsystem $\{\dot{s}_{1i} = s_{2i}, \dot{s}_{2i} = \rho_i v_i + u_{ki}, \forall t \geq t_{iF}\}$ is asymptotically stable globally in the equilibrium point.

According to the theory of finite-time stability [19], when $\rho_1 = 2 - \alpha_2$ and $\rho_2 = 1$, the subsystem is homogeneous system and the system's degree of homogeneity is $\eta = \alpha_2 - 1 < 0$. In other words, the i th subsystem $\{\dot{s}_{1i} = s_{2i}, \dot{s}_{2i} = \rho_i v_i + u_{ki}, \forall t \geq t_{iF}\}$, is globally finite-time stable. Similarly, other subsystems are globally finite-time stable and the system (see (8)) is globally finite-time stable after combination.

4. Analysis of Numerical Simulation

The effectiveness of the finite-time stable nonsmooth control algorithm based on the building structural vibration of actuator failure is verified. A three-layer building structure is simulated by three control methods including nonsmooth control, LQR control, and no control. Each floor is equipped with actuators to provide control force resisting earthquake action for structure. And the system subjected to the earthquake wave called El Centro of external disturbance signal and 15% of the actuator failure after 3 seconds is assumed. Maximum of earthquake acceleration is $a_{\max} = 3.417 \text{ m/s}^2$. The parameters of finite-time stable nonsmooth control are $\alpha_1 = 0.3$, $\alpha_2 = 0.46$, $k_1 = 4$, and $k_2 = 3$.

The mass matrix, damping matrix, stiffness matrix, and position matrix of example 1 are as follows:

$$\begin{aligned} M_1 &= 10^3 \times \begin{bmatrix} 53.2 & 0 & 0 \\ 0 & 61.3 & 0 \\ 0 & 0 & 54.5 \end{bmatrix} \text{ (kg)}, \\ C_1 &= 10^5 \times \begin{bmatrix} 32.569 & -9.0571 & 0 \\ -9.0571 & 22.666 & -7.6198 \\ 0 & -7.6198 & 13.662 \end{bmatrix} \text{ (Ns/m)}, \\ K_1 &= 10^6 \times \begin{bmatrix} 230.6 & -115.3 & 0 \\ -115.3 & 190.2 & -74.9 \\ 0 & -74.9 & 74.9 \end{bmatrix} \text{ (N/m)}, \\ E_1 &= \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & -1 & 1 \end{bmatrix}. \end{aligned} \quad (15)$$

In example 1, the contrast simulation curves of the displacement, velocity, acceleration response, and the control force for each floor are shown in Figures 1–4 under no control, LQR control, and nonsmooth control.

As is shown in Figures 1–4, nonsmooth control algorithm has been more effective than LQR control algorithm with actuator failure. The required control forces of two control methods have a little difference. However, nonsmooth control algorithm has been improved more than LQR control algorithm. In order to further analyze the effect of nonsmooth control, LQR control, and no control, the maximum displacement and maximum acceleration of each layer in the above simulation results are counted. The results are shown in Tables 1 and 2.

As is shown in Tables 1 and 2, compared with no control, the maximum displacement of the first, second, and third floor decreased by 85%, 88%, and 91% in LQR control. The maximum acceleration is also reduced by 23%, 8%, and 9%. Nevertheless, compared with LQR control, the maximum displacement of the first, second, and third floor is decreased by 74%, 77%, and 77% in nonsmooth control, respectively. And the maximum acceleration values are all reduced by 93%.

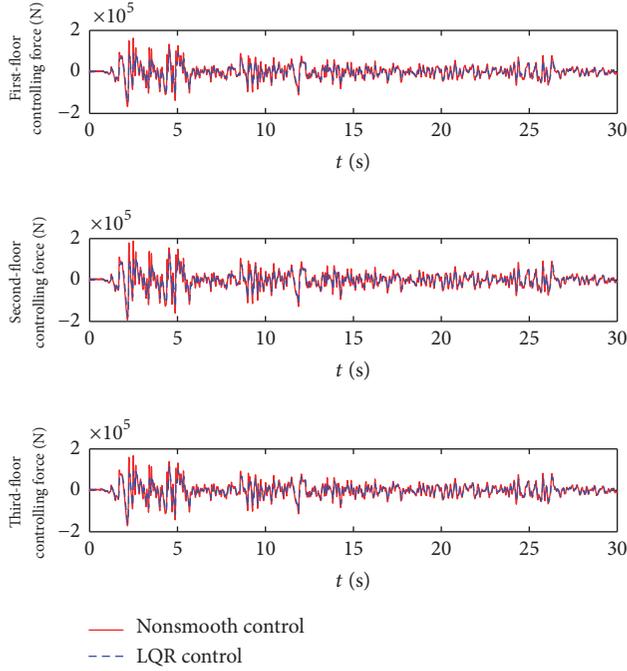


FIGURE 1: Control force of example 1.

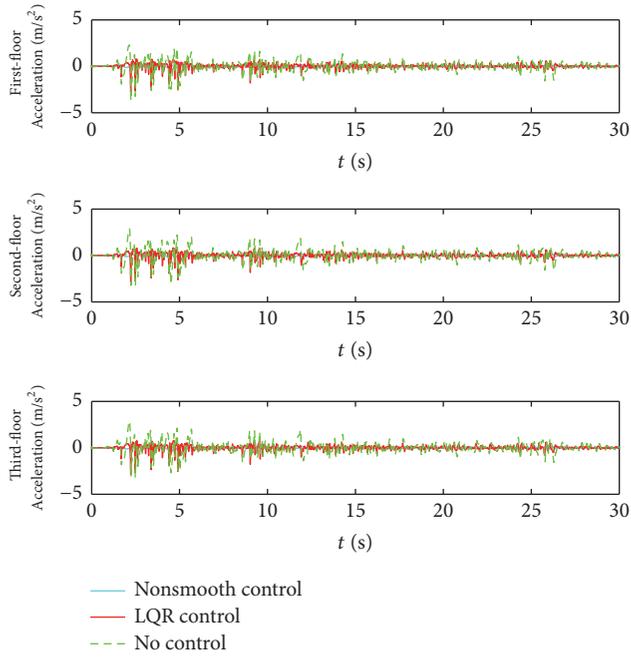


FIGURE 2: Acceleration response of example 1.

TABLE 1: Maximum displacement of each layer for example 1 (mm).

Control strategy	Nonsmooth control	LQR control	No control
First floor	1.8	6.8	44.0
Second floor	1.9	8.3	71.9
Third floor	1.9	8.3	91.3

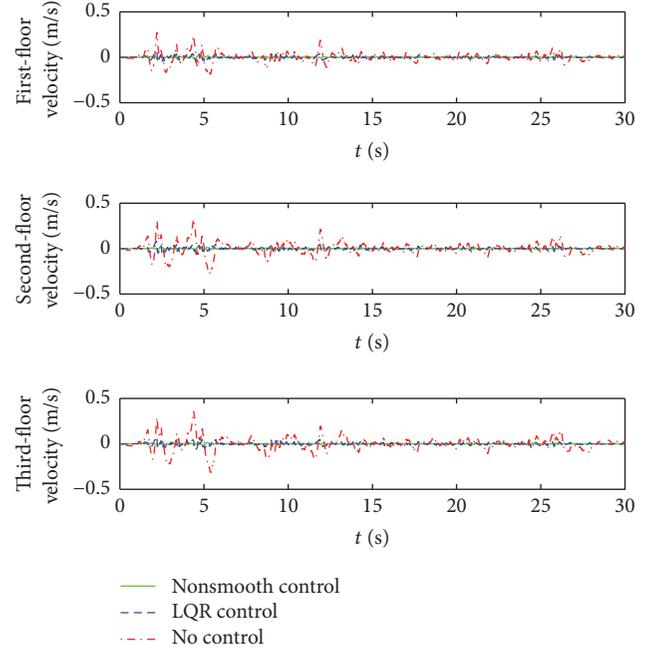


FIGURE 3: Velocity response of example 1.

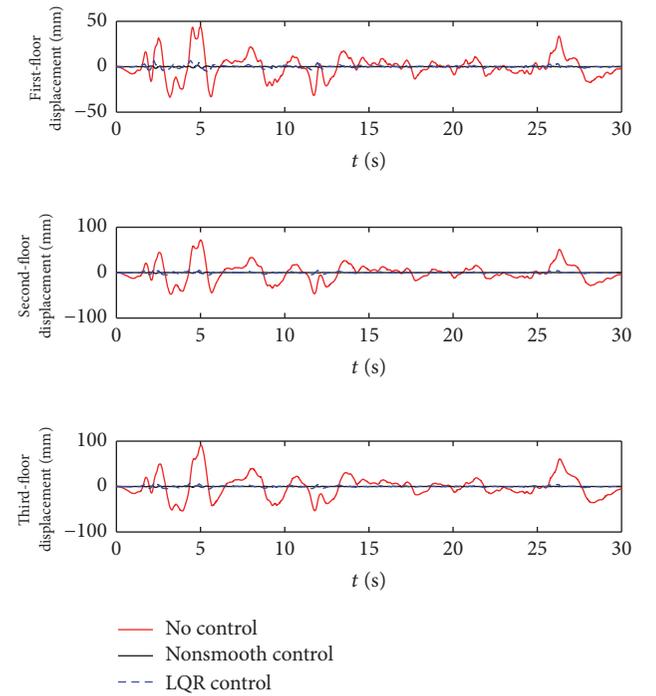


FIGURE 4: Displacement response of example 1.

TABLE 2: Maximum acceleration of each layer for example 1 (m/s^2).

Control strategy	Nonsmooth control	LQR control	No control
First floor	0.2117	2.8595	3.6929
Second floor	0.2131	3.0286	3.3022
Third floor	0.2131	2.9183	3.2125

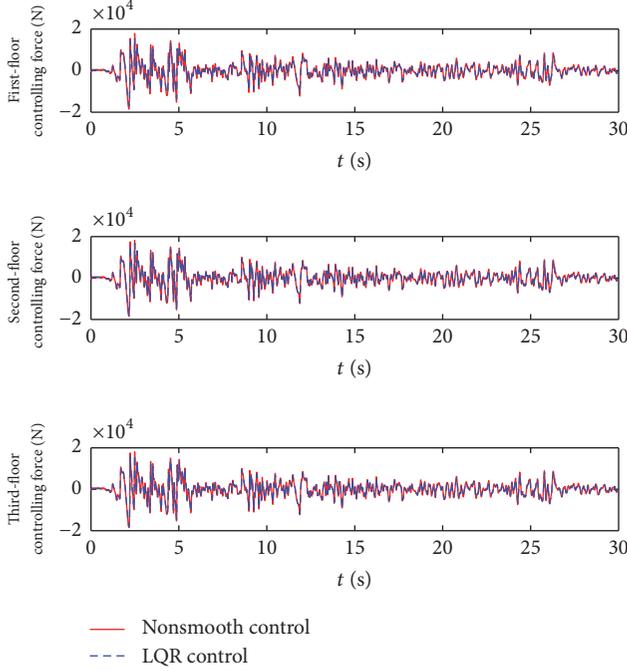


FIGURE 5: Control force of example 2.

The model parameters of example 2 are as follows:

$$\begin{aligned}
 M_2 &= \begin{bmatrix} 5897 & 0 & 0 \\ 0 & 5897 & 0 \\ 0 & 0 & 5897 \end{bmatrix} \text{ (kg)}, \\
 C_2 &= 10^3 \times \begin{bmatrix} 125 & -58 & 0 \\ -58 & 115 & -57 \\ 0 & -57 & 57 \end{bmatrix} \text{ (Ns/m)}, \\
 K_2 &= 10^3 \times \begin{bmatrix} 72825 & -39093 & 0 \\ -39093 & 67714 & -28621 \\ 0 & -28621 & 28621 \end{bmatrix} \text{ (N/m)}, \\
 E_2 &= \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ 0 & -1 & 1 \end{bmatrix}.
 \end{aligned} \tag{16}$$

In example 2, the contrast simulation curves of the displacement, velocity, acceleration response, and the control force for each floor are shown in Figures 5–8 under no control, LQR control, and nonsmooth control.

As is shown in Figures 5–8, nonsmooth control algorithm is also more effective than no control and LQR control algorithms. At the same time, the maximum displacement and acceleration of each layer from the results of the second example are counted. The results are shown in Tables 3 and 4.

As is shown in Tables 3 and 4, compared with no control, the nonsmooth control declined 96%, 98%, and 98% in the maximum displacement of the first, second, and third floor. The maximum acceleration values are all reduced

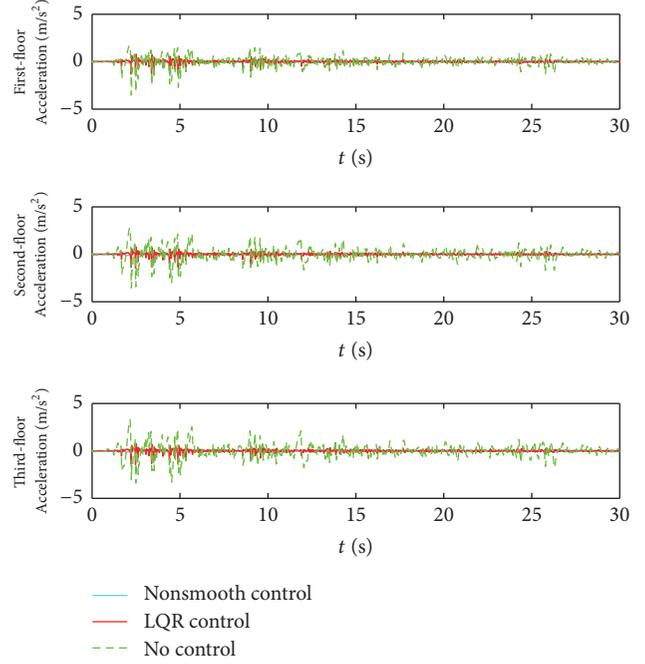


FIGURE 6: Acceleration response of example 2.

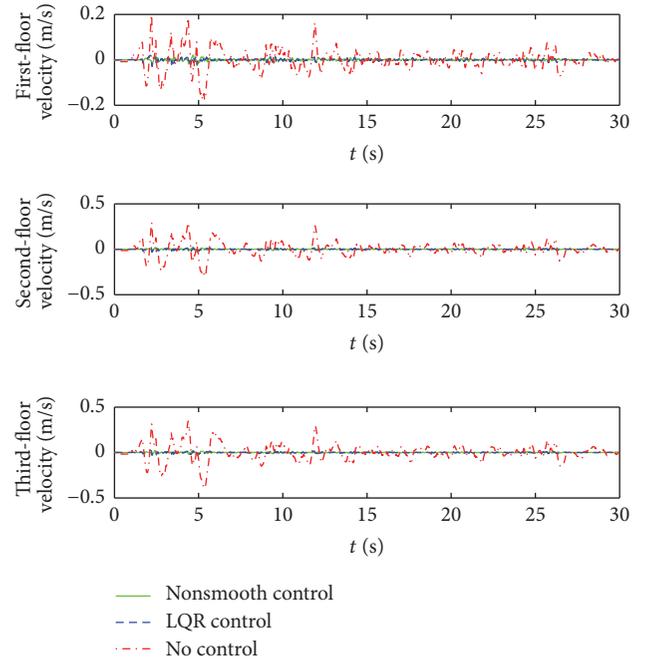


FIGURE 7: Velocity response of example 2.

by 94%. And, compared with LQR control, the maximum displacement of the first, second, and third floor is decreased by 18%, 24%, and 27% in nonsmooth control, respectively. At the same time, the maximum acceleration values are all reduced by 85%.

According to the above two examples, with the case of external distraction and actuator failure, two control methods can give good control force for displacement. However, with

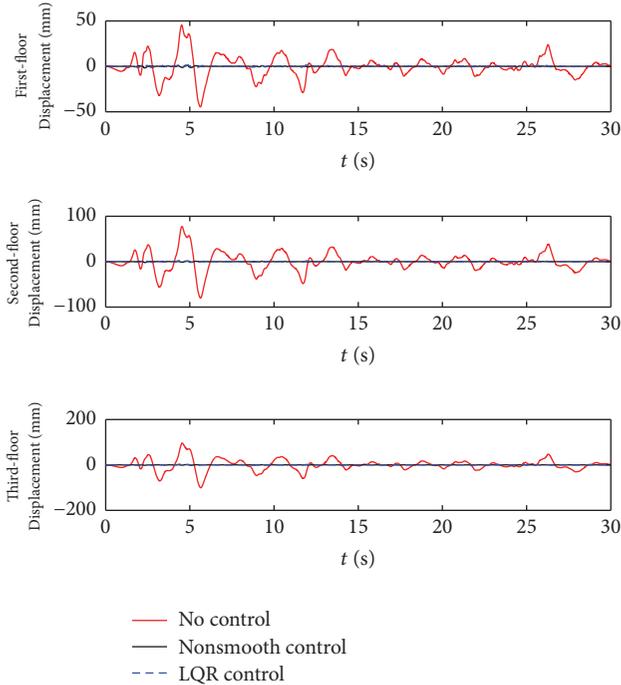


FIGURE 8: Displacement response of example 2.

TABLE 3: Maximum displacement of each layer of example 2 (mm).

Control strategy	Nonsmooth control	LQR control	No control
First floor	1.8	2.2	45.3
Second floor	1.9	2.5	80.2
Third floor	1.9	2.6	100.8

TABLE 4: Maximum acceleration of each layer of example 2 (m/s^2).

Control strategy	Nonsmooth control	LQR control	No control
First floor	0.2090	1.4286	3.5473
Second floor	0.2135	1.4604	3.5473
Third floor	0.2137	1.4598	3.3957

nonsmooth control, structural vibration is suppressed effectively better than LQR control. And interstory displacement is controlled within a small range. The displacement, velocity, and acceleration tend to a small range of vibration better and to be stable lastly. Thus, nonsmooth control algorithm can better protect the building structure from damage of the earthquake compared with LQR control algorithm.

5. Conclusion

Aiming at the problem restraining nonlinear vibration of the building structure, a structure is mathematically modeled and analyzed. Then, according to the theory of finite-time stability and the analysis of actuator failure, nonsmooth control with RBF neural network is designed for the problem of structural vibration. And the stable analysis of the system is demonstrated. Finally, nonsmooth control, LQR control,

and no control are compared by analysis. The control system is affected by seismic wave called El Centro. At the same time, the numerical simulation of the model with strong nonlinearity is studied. The above works verified the feasibility and effectiveness of nonsmooth control algorithm. In this paper, uncertainty and external perturbation estimation of the parameters are taken into account in the simulation. On this basis, further analyses of the systematic robustness and antijamming have theoretical and practical significance, which are worth studying further.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by National Natural Science Foundation (NNSF) of China under Grant no. 51478132, Guangzhou City College Scientific Research Project under Grant no. 1201630173, and Science and Technology Planning Project of Guangdong under Grant no. 2016B090912007.

References

- [1] F. Zhou, *Seismic control in engineering structures*, Seismological Press, Beijing, China, 1997.
- [2] K. Zhou, T. Wang, and J. Song, *An Introduction to Signal Detection and Estimation*, Chapter 4, Springer-Verlag, New York, NY, USA, 1985.
- [3] J. Ou, *Structural vibration control-active, semi-active and intelligent control*, Science Press, Beijing, China, 2003.
- [4] J. Wang, W. Yang, and Y. Qian, "Design of controller for torsion vibration device based on pole assignment method," *Experimental Technology and Management*, vol. 31, no. 7, pp. 86–89, 2014.
- [5] S. Tong and H. Tang, "Iterative learning instantaneous optimal control of discrete systems optimization of actuator positions," *Applied Mathematics and Mechanics*, vol. 37, no. 2, pp. 160–172, 2016.
- [6] G. Tao, S. M. Joshi, and X. Ma, "Adaptive state feedback and tracking control of systems with actuator failures," *Institute of Electrical and Electronics Engineers Transactions on Automatic Control*, vol. 46, no. 1, pp. 78–95, 2001.
- [7] X. Tang, G. Tao, and S. M. Joshi, "Adaptive output feedback actuator failure compensation for a class of non-linear systems," *International Journal of Adaptive Control and Signal Processing*, vol. 19, no. 6, pp. 419–444, 2005.
- [8] W. Gao, *Foundation of variable structure control theory*, China Science and Technology Press, Beijing, China, 1990.
- [9] L. Rosier, "Homogeneous Lyapunov function for homogeneous continuous vector fields," *Systems and Control Letters*, vol. 19, no. 6, pp. 467–473, 1992.
- [10] S. P. Bhat and D. S. Bernstein, "Finite-time stability of homogeneous systems," in *Proceedings of the American Control Conference*, pp. 2513–2514, Albuquerque, NM, USA, June 1997.
- [11] H. Hermes, "Homogeneous coordinates and continuous asymptotically stabilizing feedback controls," in *Journal of*

- Differential Equations*, vol. 127 of *Lecture Notes in Pure and Appl. Math.*, pp. 249–260, Dekker, New York, NY, USA, 1991.
- [12] K.-M. Ma, “Design of continuous non-smooth attitude control laws for spacecraft,” *The Journal of the Astronautical Sciences*, vol. 33, no. 6, pp. 713–719, 2012.
- [13] J. Wang, Q. Wang, and L. Zhang, “Design of Non-smooth Synchronous Control Method for Stage Lifting Machinery,” in *Proceedings of the 3rd International Conference on Information Science and Control Engineering (ICISCE '16)*, pp. 943–947, China, July 2016.
- [14] K.-M. Ma, “Non-smooth design and implementation of high-precision guidance laws,” *Journal of Ballistics*, vol. 25, no. 2, pp. 1–5, 2013.
- [15] J. Wang, Q. Wang, and K. Ma, “Non-smooth controller design for permanent magnet synchronous motors,” *Computer Simulation*, vol. 33, no. 3, pp. 227–230, 2016.
- [16] C. Yang, X. Wang, L. Cheng, and H. Ma, “Neural-learning-based telerobot control with guaranteed performance,” *IEEE Transactions on Cybernetics*, Article ID 2573837, pp. 1–12, 2016.
- [17] C. Yang, Z. Li, and J. Li, “Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models,” *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [18] H. Xiao, Z. Li, C. Yang et al., “Robust stabilization of a wheeled mobile robot using model predictive control based on neurodynamics optimization,” *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 505–516, 2017.
- [19] C. Yang, X. Wang, and Z. Li, “Teleoperation control based on combination of wave variable and neural networks,” *Transactions on Systems Man and Cybernetics Systems*, vol. 99, pp. 1–12, 2017.
- [20] C. Yang, J. Luo, and Y. Pan, “Personalized variable gain control with tremor attenuation for robot teleoperation,” *IEEE Transactions on Systems Man and Cybernetics Systems*, pp. 1–12, 2017.
- [21] Z. Zhao, X. Wang, C. Zhang, Z. Liu, and J. Yang, “Neural network based boundary control of a vibrating string system with input deadzone,” *Neurocomputing*, 2017.
- [22] F. Wang, B. Chen, C. Lin et al., “Adaptive neural network finite-time output feedback control of quantized nonlinear systems,” *IEEE Transactions on Cybernetics*, 2017.
- [23] J. H. Wang, Z. Liu, C. Chen, and Y. Zhang, “Fuzzy adaptive compensation control of uncertain stochastic nonlinear systems with actuator failures and input hysteresis,” *IEEE Transactions on Cybernetics*, 2017.
- [24] H. Cheng and T. Zhang, “A new predator-prey model with a profitless delay of digestion and impulsive perturbation on the prey,” *Applied Mathematics and Computation*, vol. 217, no. 22, pp. 9198–9208, 2011.
- [25] X. Dong, Z. Bai, and S. Zhang, “Positive solutions to boundary value problems of p-Laplacian with fractional derivative,” *Boundary Value Problems*, 2017.
- [26] Z. Bai, S. Zhang, S. Sun, and C. Yin, “Monotone iterative method for fractional differential equations,” *Electronic Journal of Differential Equations*, vol. 2016, article 6, 2016.
- [27] K.-M. Ma, “Design of non-smooth guidance law with terminal line-of-sight constraint,” *Journal of Ballistics*, vol. 23, no. 2, pp. 14–18, 2011.

Research Article

Structure Optimization of a Vibration Suppression Device for Underwater Moored Platforms Using CFD and Neural Network

Zhaoyong Mao and Fuliang Zhao

School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China

Correspondence should be addressed to Zhaoyong Mao; maozhaoyong@nwpu.edu.cn

Received 8 July 2017; Accepted 13 September 2017; Published 11 December 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Zhaoyong Mao and Fuliang Zhao. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We only consider the underwater mooring platform (UMP) and the plate moving in the transverse direction, and the plate can be relative to the UMP free rotation. In the case of constant flow rate ($U = 1$ m/s), the effect of different dimensionless plate length (L_p/D) and damping value (c) on the UMP was studied. We get the sample data point set by computational fluid dynamics (CFD) simulation with changing the dimensionless plate length ($L_p/D = 0.3, 0.5, 0.75, 1.0, 1.25, 1.5$) and damping value ($c = 50, 75, 100, 125, 175, 250, 300$ ($N \times s/m$)). The optimal value of the vibration suppression rate is obtained by backpropagation (BP) neural network and genetic algorithm. The optimal vibration suppression rate is $P_y = 0.9878$ and the corresponding variable value is $L_p/D = 1.0342$, $c = 57.9631$ ($N \times s/m$). In order to verify the accuracy of the optimization, we perform the CFD numerical simulation with the optimized parameters and compare the theoretical optimization results with the CFD simulation result. The absolute error between CFD simulation and optimal P_y is only 0.0037. Finally, we compare the results of CFD simulation based on optimal parameter with the bare UMP and analyze their dimensionless amplitude, wake structure, and lift coefficient. It is shown that BP neural network and generic algorithm are effective.

1. Introduction

The underwater mooring platforms (UMPs) are a class of underwater devices and with an anchor link to work in the sea. UMPs have broad application in both civil and military missions. Pressure, water quality detection, camera and acoustic communication, and other sensors are installed on the platform. It can be used for marine environmental management, resource protection, disaster monitoring, and marine hydrological monitoring in civil areas, and it also can be used for our army surface ships and submarines for remote reconnaissance defense. UMPs require as much as possible to stabilize and impact resistance to ensure that a variety of sensors can work properly. Therefore, in order to improve the stability of the UMP, it is very meaningful to study the vibration suppression control technology of UMP. At present, the research on the characteristics of the UMP is mainly focused on the simulation of the anchor chain and the cable, while the research on the stability of the UMP is

especially less. Research on the underwater vibration control is mainly concentrated in the field of offshore riser.

The suppressing method of vortex-induced vibration can be divided into two types: active control and passive control measures. Active control measures are real-time monitoring of flow field changes and structural forces. It is through the automatic technology to disturb the flow field, which can control the cylinder vortex shedding and finally achieve the purpose of vibration suppression. Passive control measures are to change the flow field by directly modifying the external shape of the structure or other devices attached to the structure. It is to achieve the purpose of controlling the formation and shedding of the vortex. Compared with the active control measures, passive control measures are simple to design, easy to manufacture, and low cost. So it has been widely used in the field of marine engineering.

In this paper, taking into account the stability of the UMP, we have adopted a relatively simple vibration suppression

device-splitter plate which can rotate freely relative to the UMP.

In the previous research literatures, there are many studies of the cylinder with the fixed splitter plate and less research on the splitter plate that is free to rotate behind the cylinder. The majority is to study the effect of fixed dividers on vortex shedding. For the elastic-supported cylinder-splitter plate system, Ding et al. [1] studied the wake characteristics and vortex-induced vibration characteristic of the splitter plate under high Reynolds number, where the length of the splitter plate is the same as the diameter of the cylinder. The results show that, with high Reynolds number, the amplitude of the cylinder with plate does not decrease (without the effect of suppressing vibration) but increases as the number of Reynolds increases. The vibration frequency is lower than the bare cylinder and the vortex shedding model of cylinder with a splitter plate and a bare cylinder is also different. The inhibitory effect is more obvious as the velocity decreases. Tan et al. [2] carried out a two-dimensional numerical simulation with a splitter plate cylinder and a bare cylinder, and the ratio of the length of the splitter plate to the diameter of the cylinder varies from 0.25 to 2.0 ($L/D = 0.25\sim 2.0$). The results show that the splitter plate has the most significant effect on the vortex-induced vibration control of the riser when L/D is 1.0–2.0. Wang et al. [3] performed a two-dimensional computational fluid dynamics (CFD) numerical simulation of the fixed separation plate (variable dimensionless length). When the Reynolds numbers are 1000 and 30000, respectively, changing $L/D = 0.5\sim 2.0$ to simulation can be found to reduce the drag coefficient, lift coefficient, and vortex shedding frequency. This document of Amiraslanspour et al. [4] is used to study the two splitter plate in the upstream or downstream stages of the cylinder. They found the influence of the splitter plate on the drag force of the cylinder. They also studied the effect of the distance from the end of the splitter to the cylindrical surface without much impact on the resistance. When the ratio of amplitude to the cylinder diameter (A/D) was 0.25 and 0.5, respectively, the resistance was reduced by 57% and 36%. Qiu et al. [5] experimentally studied the fixed cylinder with frontal plate, bare cylinder, cylinder with rear splitter plate, cylinder with bilateral plates, and semicylindrical roof. They found that the cylinder with wake splitter plate ($L/D = 3$) can effectively inhibit the vortex shedding, and shear layer was separated on both sides of the plate. Lou et al. [6] experimentally studied the effect of vortex-induced vibration of two adjacent risers with splitter plates and found that the splitter plate on the risers was effectively in suppressing both cross-flow (CF) and in-line (IL). Assi et al. [7] experimented with plain cylinder, cylinder with fixed splitter plate, cylinder with free-to-rotate splitter plate, and cylinder with pairs of plates and studied their vibration amplitude at different reduced speed. The results show that these devices can effectively reduce the resistance and suppress vortex vibration. Assi et al. [8] have mainly studied the stability of a free-to-rotate short-tail fairing and a splitter plate as suppressors of vortex-induced vibration. They studied the rotating (free-to-rotate suppressors in 2-dof) and nonrotating (fixed suppressors in 1-dof) at different reduced speed. It is found that the free-rotating suppressors

will be deflected position to remain stable and extend the vortex shedding and suppressing vortex-induced vibration. It is also found that fixed suppressors will show serious gallop in a considerable flow range of flow speeds. Gozmen et al. [9] and Zhu et al. [10] studied the relationship of different lengths of splitter plate to suppress vortex shedding. Gu et al. [11] studied a fixed cylinder with a free splitter plate, where the ratio of the length of plate to the diameter of the cylinder varied from 0.5 to 6 and the Reynolds number varied from 30000 to 60000. The pressure distribution, drag, lift force, and vortex shedding model are analyzed, and the study finds that the free rotation angle corresponding to the length of the longer plate is smaller. Akilli et al. [12] studied that the influence of factors such as the dimensionless thickness (T/D) on the splitter plate and the distance (G/D) between the end of the plate and the fixed cylinder vortex shedding was studied. The dimensionless thickness of the plate varies from 0.016 to 0.08, and the distance is changed from 0 to 100 (mm) in the step size of 12.5. When G/D changed from 0 to 1.75, the plate has effect on suppression of the vortex shedding. Law and Jaiman [13] studied four kinds of vibration-proofing devices (fairing, connected-C, disconnected-C, and splitter plate) separately, which the device can not rotate freely relative to the cylinder. It was found that the connected-C device was similar to the effect of the fairing device on vortex-induced vibration. The study also found that the connected-C device and disconnected-C device can effectively prevent the cylinder galloping at high reduced velocity, but the connection between them has little effect on vibration suppression. In [14], the authors mainly studied the circular cylinders fitted with three different geometries of splatter plate (solid splitter plate $L/D = 0.5$, solid splitter plate $L/D = 1.0$, and slotted splitter plate $L/D = 1.0$). He studied the response of the different vibration mechanisms of the three devices with varying reduced speeds.

With reference to the above-mentioned latest literature, there are few studies on the splitter plate parameters and the splitter plate which are rotated relative to the cylinder. So in this paper, I mainly studied the effect of the length of the splitter plate that can rotate freely around the cylinder and the damping values between the splitter plate and the cylinder on the vibration suppression technique of the UMP.

For some complex numerical simulation, if we all use CFD simulation solution, then we will greatly increase the calculation of cost and the calculation cycle. Therefore, in order to improve the computational efficiency, this paper establishes an optimization method based on the combination of CFD and backpropagation (BP) neural network and genetic algorithm. BP neural network and genetic algorithm are used at some of the nonlinear data to find the optimal value [15–17]. Adaptive control [18, 19] in the neural network has also been a very good application. Safikhani et al. [20] have studied the multiobjective optimization of nanofluid flow in flat tubes, combined with a CFD, artificial neural networks, and genetic algorithms. They got important design information about nanofluids and flat tubes by combining CFD, GMDH, and THE multiobjective optimization method. Avci et al. [21] studied the optimization of the design parameters of a home refrigerators using CFD and artificial neural

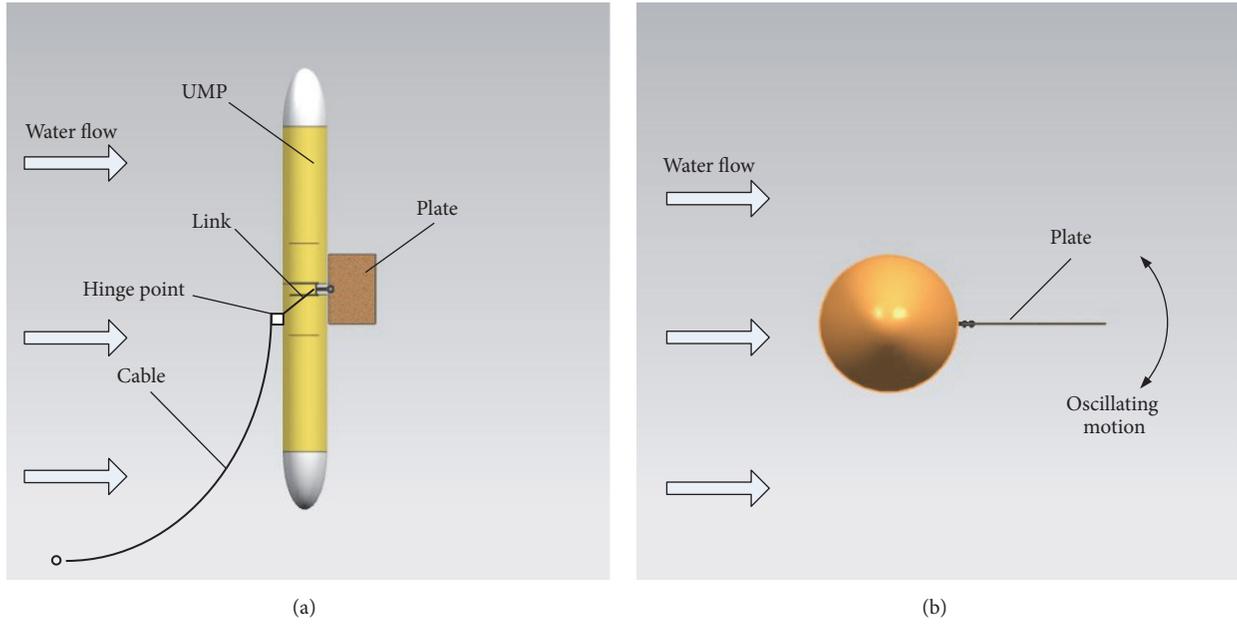


FIGURE 1: (a) Overall diagram of the UMP. (b) A schematic diagram of the splitter plate for rotation.

network. This paper shows that CFD simulation and the ANN can determine the best value for the refrigerator design.

So we can combine CFD simulation and BP neural network genetic algorithm to optimize the best vibration suppression rate. Through the use of CFD simulation software, we can get some data about vibration suppression rate, finally combined with BP neural network genetic algorithm to find the best suppression effect and the corresponding dimensionless plate length and damping value size.

2. Geometry Configuration

One end of the cable is tethered on the UMP and the other end is fixed under the sea, so that the UMP is floating in the sea. Some sensor can not work properly, because the ocean flow easily leads to the vortex-induced vibration. Inspired from the above research, we installed a splitter plate in the wake area of the UMP, in which the splitter plate can freely rotate around the UMP. As shown in Figure 1, (a) is the overall diagram of the UMP, and (b) is a schematic diagram of the splitter plate for rotation. The vortex shedding process alternates on both sides of the UMP and will produce vortex-induced vibrations because of the UMP generated periodic pulsating forces. So, we designed a rotatable splitter plate to extend the process and achieve the effect of vibration suppression. In this paper, we mainly focus on evaluating the wake structure, force, motion (the UMP only moves in the transverse direction with a degree of freedom, and the splitter plate is rotated relative to UMP), and the transverse amplitude at different design parameters with a CFD method.

Because the two-dimensional model requires less grid and shorter calculation time, the general vortex-induced vibration numerical simulation is based on two-dimensional numerical simulation. A simplified two-dimensional (2D)

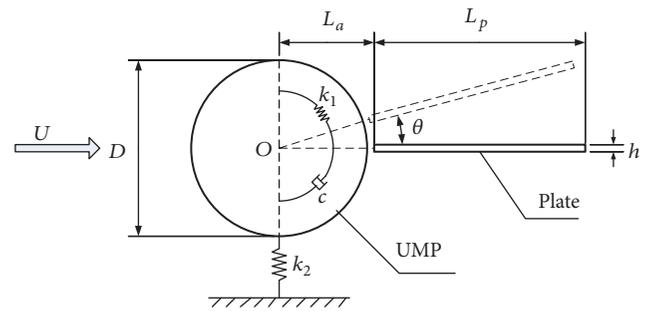


FIGURE 2: A simplified 2D model of UMP with a splitter plate.

model of UMP with a splitter plate is shown in Figure 2. The parameters indicated in Figure 2 that U represents the velocity of the upstream flow; L_p represents the length of the plate; L_a represents the distance between the root of the plate and the center of the UMP; θ represents the rotation angle of the plate; k_1 represents the spring stiffness between the plate and the UMP; c represents system damping; D represents diameter of the UMP and h represents the height of the plate; k_2 represents the simplified spring stiffness between the UMP and the cable. Where k_2 is derived from (1), $k_2 = 1599$ (N/m) value is the constant value in this paper.

$$Fd \cdot l \cdot \sin(\alpha) = k_2 \cdot \alpha \quad (1)$$

$$k_2 = Fd \cdot l.$$

In this paper, we set the distance between the center of the UMP and the root of plate $L_a = 0.3$ (m) and the diameter of the cylinder is $D = 0.533$ (m). A typical flow velocity of $U = 1$ m/s is selected for all simulations in this paper.

The specific values of each parameter are shown in Table 1.

TABLE I: Parameters in simulation.

Parameter	D	U	L_a	L_p	k_1	k_2	h	c
Value	0.533 m	1 m/s	0.3 m	~	~	1599 N/m	0.002 m	~

Note. “~” indicates that the value is variable.

2.1. Governing Equations and Turbulence Model. In this paper, the numerical simulation of the calculation medium is incompressible viscous fluid and does not take into account the temperature changes. So it belongs to the unsteady flow. The governing equations include continuity equations and momentum equations, as shown as follows:

$$\begin{aligned} \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} &= 0 \\ \frac{\partial u_x}{\partial t} + u_x \frac{\partial u_x}{\partial x} + u_y \frac{\partial u_x}{\partial y} &= -\frac{1}{\rho} \frac{\partial p}{\partial x} + \nu \left(\frac{\partial^2 u_x}{\partial x^2} + \frac{\partial^2 u_x}{\partial y^2} \right) \\ \frac{\partial u_y}{\partial t} + u_x \frac{\partial u_y}{\partial x} + u_y \frac{\partial u_y}{\partial y} &= -\frac{1}{\rho} \frac{\partial p}{\partial y} + \nu \left(\frac{\partial^2 u_y}{\partial x^2} + \frac{\partial^2 u_y}{\partial y^2} \right), \end{aligned} \quad (2)$$

where u represents the velocity of the fluid in the i direction; ρ represents the fluid density; ν represents the kinematic viscosity coefficient of the fluid; i, j represent the direction of the different axes.

At present, the turbulence model commonly used in computational fluid dynamics mainly includes Reynolds stress model and viscous vortex model. The Reynolds stress model is a direct complement to the equation that expresses the Reynolds stress equation to solve the time-averaged motion control equation. The vortex model is the assumption that the turbulence viscosity and the vorticity coefficient are introduced to supplement the Reynolds stress tensor $-\rho \overline{u_i u_j}$ and (3) can be seen.

$$-\rho \overline{u_i u_j} = \mu_t \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - \frac{2}{3} \left(\rho k + \mu \frac{\partial U_i}{\partial x_i} \right) \delta_{ij}, \quad (3)$$

where μ_t represents turbulent vortex viscosity coefficient; k represents turbulent kinetic energy, in which

$$k = \frac{1}{2} \overline{u_i u_j} = \frac{1}{2} \left(\overline{u_x'^2} + \overline{u_y'^2} + \overline{u_z'^2} \right). \quad (4)$$

The SST $k-\omega$ model combines the advantages of the $k-\epsilon$ model and the $k-\omega$ model, namely, the advantages of the boundary layer problem and the superior field calculation. Using the SST $k-\omega$ turbulence model to numerically simulate the vortex-induced vibration problem is more accurate, so this paper uses the SST $k-\omega$ model to carry out numerical simulation.

2.2. Equations of Motion. The dynamic equation of the plate which can freely rotate around the UMP can be obtained by a typical mass-spring-damper system and be presented as

$$J \ddot{\theta}(t) + c \dot{\theta}(t) + k_1 \theta(t) = M(t), \quad (5)$$

where J is the total rotational inertial of the plate; c is the system damping, k_1 is the stiffness of the spring, θ is the oscillating angle of the plate, and $M(t)$ is the hydrodynamic force on the plate.

The total rotational inertial (J) is the sum of the plate structure rotational inertial (J_b) and the added mass component (J_a). J_b and J_a can be calculated by following equation, respectively:

$$J = J_a + J_b \quad (6)$$

$$J_b = \frac{1}{3} \rho_{\text{plate}} h L_{\text{feature}} \left[(L_a + L_p)^3 - L_a^3 \right] \quad (7)$$

$$J_a = \frac{1}{16} \pi \rho_{\text{water}} L_{\text{feature}} \left[(L_a + L_p)^4 - L_a^4 \right], \quad (8)$$

where L_{feature} is the feature length of the UMP and plate, $L_{\text{feature}} = 1$ m; ρ_{plate} is the density of the plate; h is the height of the plate; L_a represents the distance between the root of the plate and the center of the UMP; L_p represents the length of the plate; ρ_{water} is the density of the water.

The vibration equation (1-dof) of the UMP only in the transverse direction can be obtained by the mass-spring stiffness system and be presented as

$$M \ddot{y}(t) + k_2 y(t) = F_l(t), \quad (9)$$

where M is the total mass of the UMP which is sum of the total mass of the UMP (M_1) and the total mass of the plate (M_2). The above M_1 is the sum of the additional mass of the UMP ($M_{\text{add(UMP)}}$) and the mass of the UMP itself ($M_{\text{itself(UMP)}}$); M_2 is the sum of the additional mass of the plate ($M_{\text{add(Plate)}}$) and the mass of the plate itself ($M_{\text{itself(Plate)}}$). Since the angle of the freely rotating plate is small, the additional mass of the plate ($M_{\text{add(Plate)}}$) is approximately calculated as a cylinder with a diameter equal to its length. As shown in formula (10). k_2 is the equivalent stiffness between the cable and the UMP, which can be obtained from formula (1). F_l is the total lift of the whole system that is the sum of the cylinder lift of the cylinder lift and plate lift. y , \dot{y} , and \ddot{y} are the displacement, velocity, and acceleration in the y direction (transverse vibration direction).

$$M = M_1 + M_2$$

$$M_1 = M_{\text{add(UMP)}} + M_{\text{itself(UMP)}}$$

$$M_{\text{itself(UMP)}} = 200 \text{ kg}$$

$$M_{\text{add(UMP)}} = \pi \times \left(\frac{D}{2} \right)^2 \times L_{\text{feature}} \times \rho_{\text{water}} \times C_p \quad (10)$$

$$M_2 = M_{\text{add(plate)}} + M_{\text{itself(plate)}}$$

$$M_{\text{itself(plate)}} = L_p \times h \times L_{\text{feature}} \times \rho_{Al}$$

$$M_{\text{add(plate)}} = \pi \times \left(\frac{L_p}{2} \right)^2 \times L_{\text{feature}} \times \rho_{\text{water}} \times C_p,$$

where L_{feature} is the feature length of the UMP and plate, $L_{\text{feature}} = 1$ m; ρ_{water} is density of the water, $\rho_{\text{water}} =$

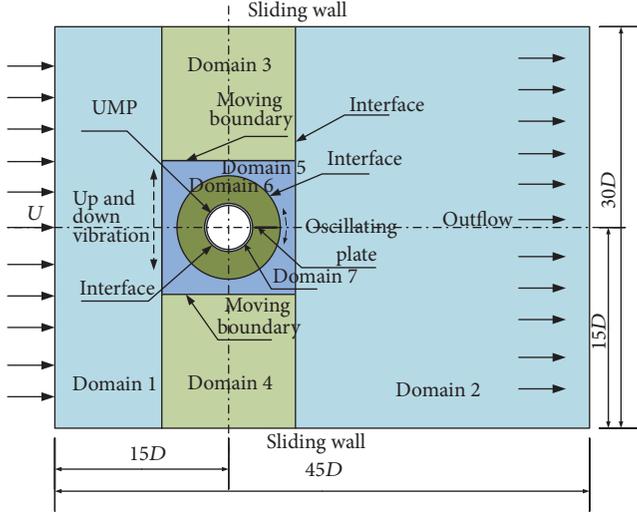


FIGURE 3: Computational domain and boundary conditions.

1000 kg/m^3 ; C_p is an additional mass factor, $C_p = 1$; ρ_{Al} is the density of the plate, $\rho_{Al} = 2700 \text{ kg/m}^3$.

In order to eliminate the dimensional effects, the lift, drag, and torque are also normalized, and the natural frequency of the plate is shown in (14).

$$C_l = \frac{F_l}{(1/2) \rho_{\text{water}} D H U^2} \quad (11)$$

$$C_d = \frac{F_d}{(1/2) \rho_{\text{water}} D H U^2} \quad (12)$$

$$C_m = \frac{M}{(1/4) \rho_{\text{water}} D^2 H U^2} \quad (13)$$

$$f = \frac{1}{2\pi} \sqrt{\frac{k_1}{J}}. \quad (14)$$

3. Numerical Approach

3.1. Computational Domain and Boundary Conditions. A rectangle with a length of $45D$ and a width of $30D$ is used as the computational domain of this paper. As shown in Figure 3, it can be seen from the figure that the center of UMP is placed at a distance from the left boundary of the computational domain to $15D$ and on the symmetry axis of the top and bottom boundary. The overall domain is split into seven subdomains, including two external static domains, two dynamic domains for updating the grid with layering method [22, 23], two inner stationary domains 5 and 6, and an oscillating domain. This inner stationary domain 7 contains the cells around the UMP. The rotation domain where the plate is located is divided by two sliding interfaces in two inner stationary domains (domain 5 and domain 7). The stationary domain 5 has two upper and lower moving boundaries for the grid update. The grid updating domain (domain 3 and domain 4) and the inner entire UMP motion domain (domain 5, domain 6, and domain 7) are divided

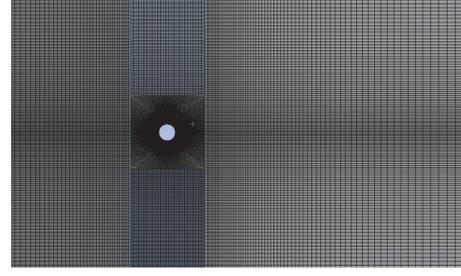


FIGURE 4: The mesh graph for the entire computational domains.

by an external stationary domain (domain 1 and domain 2). The following is a description of the boundary conditions: (1) inlet: set the uniform and steady flow of incoming speed 1 m/s ; the corresponding Reynolds number is 533000 ; (2) outlet: outflow exit boundary conditions; (3) interface: the overlapped edges between the seven domains are set as interfaces; (4) wall: the surface of the UMP and the plate are set to no slip boundary conditions, whereas the top and the bottom edges are set to slip boundary conditions.

3.2. Mesh Generation. The entire computational domains are a quadrilateral mesh cell. It is worth nothing here that the rigid boundary layer must be added to the surface of the UMP and plate which the boundary layer grid moves with the UMP and plate and is stable and the shape no longer changes to provide sufficient precision. The first layer height of the boundary layer is determined according to the value of y^+ which is suggested by Fluent User's Guide [24] for the use of the SST $k-\omega$ turbulence model. So here set the first layer height as 0.0002 m , and UMP and plate boundary layer were set to 16 layers and 12 layers. Figure 4 shows the mesh graph for the entire computational domains. Figure 5 shows a mesh diagram of UMP and plate. The oscillating domain rotates about the center of the UMP and rotates along the interfaces, and the mesh of the rotation domain does not change and update.

3.3. Mesh Independence, Numerical Method, and Time Step Validation

3.3.1. Mesh Independence Validation and Numerical Method Validation. Using the CFD numerical simulation, the meshing and time step have a significant effect on the results. A mesh independence verification study is conducted using three grids with different nodes densities ($41000/66900/97000$ elements). The mesh verification is carried out by means of a fixed cylinder and not rotating the plate ($k_1 = \infty$, $k_2 = \infty$, $c = \infty$, and $Lp = D$). The independence of the mesh is verified by assessing the root mean square (RMS) of the lift coefficient and the mean drag coefficient for different mesh densities. The results are summarized in Table 2. Both C_l and C_d have the good consistency for different mesh densities. Accordingly, the mesh with medium density can predict the performance with sufficient accuracy and will be used in the later simulations.

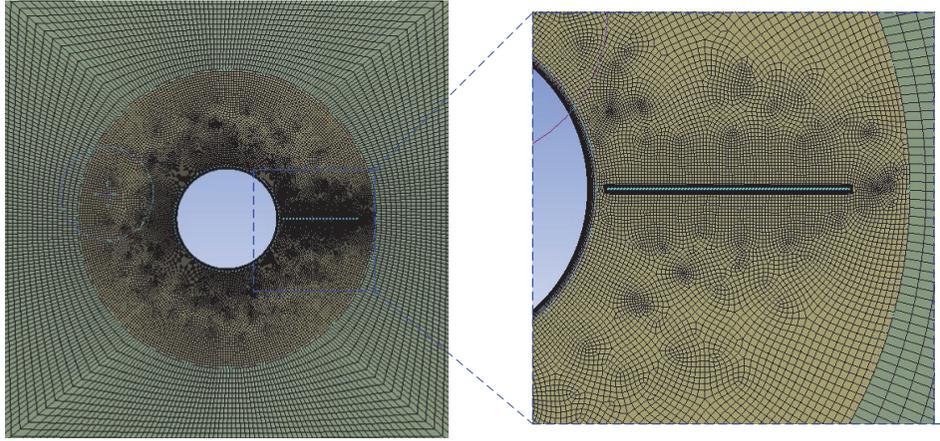


FIGURE 5: Mesh diagram of UMP and plate.

TABLE 2: Mesh convergence results.

Mesh	Number of elements	RMS of Cl	Averaged Cd
(1)	41000	0.145321	0.515543
(2)	66900	0.162324	0.571632
(3)	97000	0.172251	0.583214

TABLE 3: Physical model parameters for the simulation.

Description	Symbol	Value
Mass ratio	m^*	0.93
Damping ratio	ξ_{total}	0.1076
Velocity ratio	U^*	3–13
Spring stiffness per unit length	K (N/m)	76.5
Diameter	D (mm)	32

Some of the previous experiments have been done on the study of the splitter plate in the wake of a cylinder. In order to prove the validity of the numerical method, we use CFD to simulate a 1-degree-of-freedom vortex-induced vibration with a cylinder and then compare the numerical results with the experimental data. The testing parameters adopted in the validation simulations are the same with the experiments and are listed in Table 3.

Figure 6 shows the comparison of the amplitude and frequency response between the numerical and the experimental results. It is observed that the VIV amplitudes obtained from the numerical method are in good agreement with the experimental data and the lock-in of amplitude (corresponding to the upper branch) is well characterized, except a slightly overestimate at high flow velocities. As for the frequency, the numerical results coincident with the experimental results well. Therefore, comparison with experiments shows that the numerical method is quite credible and acceptable.

3.3.2. Time Step Validation. In this paper, the numerical simulation parameters of the time step verification are the same as the simulation parameters of this paper. We just

TABLE 4: Different time steps of the RMS of lift and averaged drag force.

Time step	RMS of Cl	Averaged Cd
0.001	0.169321	0.574532
0.002	0.168376	0.573810
0.005	0.162324	0.571632
0.01	0.150719	0.567762
0.02	0.127477	0.559646

simulate the flow of the fixed cylinder with a plate to verify the time step. We carry out the numerical simulation with the dimensionless plate length 1. The time step is 0.001, 0.002, 0.005, 0.01, and 0.02. We validate the effective time step by referring to the root mean square (RMS) and the mean drag coefficient at different time steps. This will not only get accurate results but also improve the efficiency of computing. The results are summarized in Table 4. The lift coefficient curve is shown in Figure 7. Table 4 and Figure 7 show that, in the premise of ensuring the accuracy of the calculation, while improving the efficiency of the calculation, we choose 0.005 as the time step of this paper.

4. Numerical Simulation

4.1. Numerical Simulation Conditions. In the whole numerical simulation of this paper, we mainly study the effects of plate length and damping (damping between UMP and plate) variation on the UMP amplitude. In addition, the thickness of the plate and the front of the plate to the center of the UMP (L_a) also have a certain impact on the vibration suppression rate. In order to reduce the amount of calculation, we set the plate thickness and distance (L_a) as a constant. Other related numerical simulation parameters are f ($f = 0.5$, natural frequency of the plate), h ($h = 0.002$ m, height of the plate), and $k_2 = 1599$ (N \times s/m) (as can be seen from (1)). Equations (6), (7), (8), and (14) show that when we change the plate length, the total moment of inertia of the plate also changes with (6), (7), and (8). The spring stiffness of the plate changes

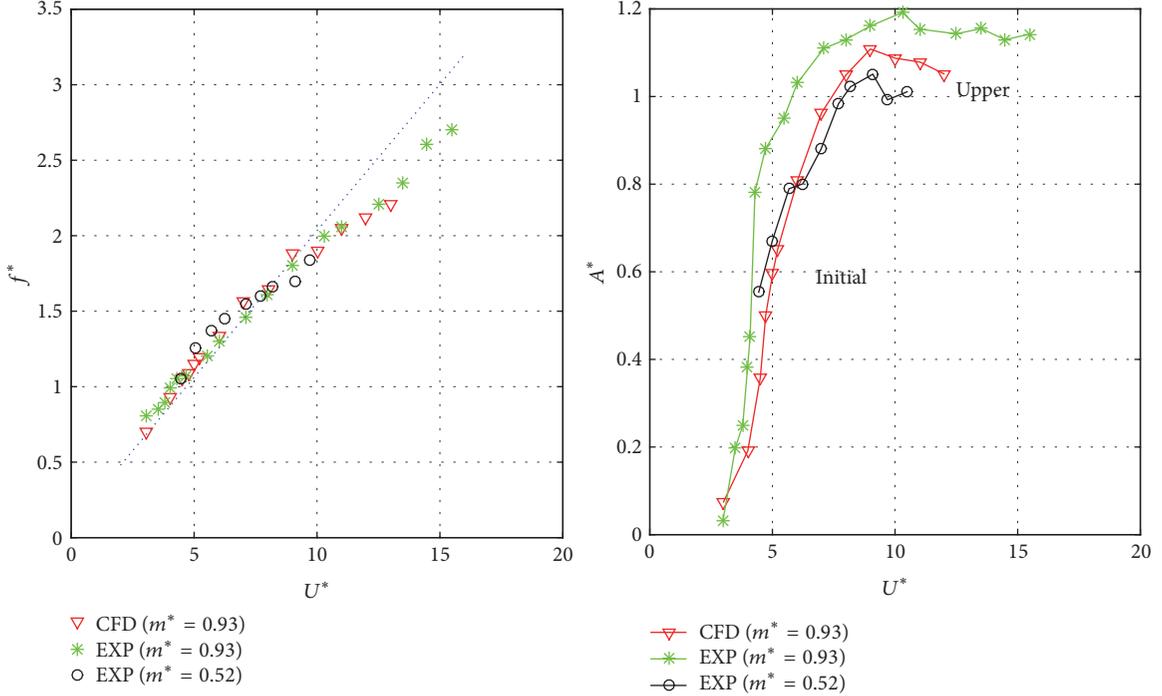


FIGURE 6: Comparison of amplitude and frequency responses between the numerical results and the experimental results.

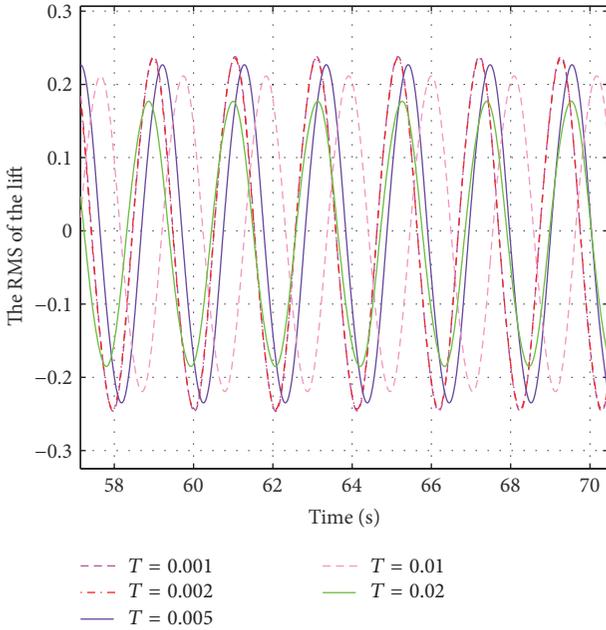


FIGURE 7: The lift coefficient of different time steps.

with the total moment of inertia of the plate is also obtained by (14). All case calculations, the density of the mesh, and time steps are all the same. Here we chose the plate length data $L_p/D = 0.3, 0.5, 0.75, 1.0, 1.25, 1.5$ and damping data 50, 75, 100, 125, 175, 250, and 300 ($N \times s/m$). The range of each variable is shown in Table 5.

TABLE 5: Design variables and their range of variations.

Design variables	From	To	Selected
L_p/D	0.3	1.5	0.3, 0.5, 0.75, 1.0, 1.25, 1.5
c ($N \times s/m$)	50	300	50, 75, 100, 125, 175, 250, 300

4.2. Numerical Simulation Results. In this paper, the influence of dimensionless plate length (L_p/D) and damping (c) on the vibration of UMP is studied under the condition that the other parameters remain unchanged. We firstly simulate the vortex-induced vibration of a bare cylinder under the same computational simulation conditions. We compare the vibration amplitude, lift coefficient, and drag coefficient of the bare cylinder with the cylinder with the plate. The parameter of the study, the amplitude of the vibration (A), the dimensionless amplitude (A_y^*), and the suppression rate P_y are shown in Table 6. Here it is the dimensionless amplitude A_y^* which is the ratio of the actual amplitude to the diameter of UMP.

Through Table 6, study found that when the dimensionless plate length (L_p/D) is 0.3 and the damping value (c) is greater than 75 ($N \times s/m$), the UMP does not show the effect of vibration suppression but increased the vibration amplitude.

From the finite discrete parameter data point, we can see that the amplitude of the UMP is the smallest when the dimensionless plate length (L_p/D) is 1.25, the damping (c) is 175 ($N \times s/m$), and the maximum suppression rate (P_y) is 0.979676. The effect on the suppression vibration achieves the best results.

TABLE 6: Numerical simulation parameter data display table.

Number	L_p/D	c (N \times s/m)	A (m)	A_y^*	P_y
(1)	—	—	0.451631	0.847339	—
(2)	0.3	50	0.386184	0.724547	0.144914
(3)	0.3	75	0.451842	0.847734	-0.00047
(4)	0.3	100	0.487460	0.914559	-0.07933
(5)	0.3	125	0.524511	0.984073	-0.16137
(6)	0.3	175	0.543321	1.019364	-0.20302
(7)	0.3	250	0.522656	0.980593	-0.15726
(8)	0.3	300	0.512464	0.961472	-0.1347
(9)	0.5	50	0.289810	0.543733	0.358304
(10)	0.5	75	0.312187	0.585717	0.308757
(11)	0.5	100	0.338961	0.635948	0.249475
(12)	0.5	125	0.367375	0.689259	0.18656
(13)	0.5	175	0.434383	0.814977	0.038191
(14)	0.5	250	0.447285	0.839185	0.009622
(15)	0.5	300	0.448789	0.842005	0.006293
(16)	0.75	50	0.011683	0.021920	0.974131
(17)	0.75	75	0.010332	0.019384	0.977124
(18)	0.75	100	0.012791	0.023999	0.971677
(19)	0.75	125	0.015718	0.029490	0.965197
(20)	0.75	175	0.020008	0.037539	0.955698
(21)	0.75	250	0.148309	0.278252	0.671616
(22)	0.75	300	0.147588	0.276900	0.673212
(23)	1.0	175	0.010143	0.019031	0.97754
(24)	1.0	250	0.015051	0.028238	0.966674
(25)	1.0	300	0.016659	0.031256	0.963113
(26)	1.25	50	0.015277	0.028662	0.966174
(27)	1.25	75	0.012806	0.024026	0.971645
(28)	1.25	100	0.010937	0.020519	0.975784
(29)	1.25	125	0.009723	0.018242	0.978471
(30)	1.25	175	0.009179	0.017222	0.979676
(31)	1.25	250	0.010298	0.019320	0.977199
(32)	1.25	300	0.012559	0.023562	0.972192
(33)	1.5	50	0.030614	0.057437	0.932215
(34)	1.5	75	0.028877	0.054178	0.936061
(35)	1.5	100	0.027484	0.051565	0.939145
(36)	1.5	125	0.026268	0.049283	0.941838
(37)	1.5	175	0.024477	0.045924	0.945802
(38)	1.5	250	0.022894	0.042952	0.949309
(39)	1.5	300	0.021951	0.041185	0.951395

5. Optimal Design Method

For the unknown nonlinear function, it is difficult to accurately find the extremum of the function only through the input and output data of the function. This kind of problem can be solved by BP neural network and genetic algorithm. The nonlinear fitting ability of BP neural network and the nonlinear optimization ability of genetic algorithm are used to find the extreme value. For some complex numerical simulation, if we all use CFD simulation solution, then we will greatly increase the calculation of cost and the calculation cycle. Therefore, in order to improve the computational

efficiency, this paper establishes an optimization method based on the combination of CFD and BP neural network and genetic algorithm. The optimization method is to obtain the sample point of the data by CFD simulation and then use BP neural network to establish the agent model between the influencing factors and the response value. Then the genetic algorithm is used to solve the optimization model, and the optimal problem is obtained solution.

5.1. BP Neural Network. BP neural network is a kind of multilayer feedforward network with error backpropagation, which has very strong nonlinear mapping approximation

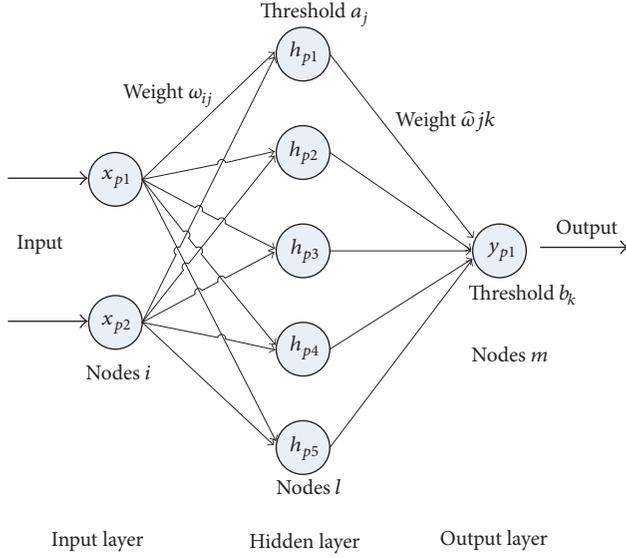


FIGURE 8: The topology of BP neural network.

ability and is the most widely used neural network. BP neural network is mainly through the input data for network training, finally establishes a mapping relationship, and then does network prediction output. The neural network is widely used, for example, in the motion control of an underactuated wheeled inverted pendulum model [25] and robotic self-identification for surrounding obstacle [26]. The topology of BP neural network is shown in Figure 8. The structure has two input layers nodes, five hidden layers nodes, and one output layer node. Each node of the structure represents a neuron, and the nodes between the layers and the layers are connected by weights. The following is the working principle of BP neural network.

(1) *Network Initialization.* According to CFD simulation results, determine the network input data and output data (X , Y). H is the data matrix of the hidden layer, as can be seen in

$$\begin{aligned} X_p &= (x_{p1}, x_{p2}, \dots, x_{np}) \\ H_p &= (h_{p1}, h_{p2}, \dots, h_{pl}) \\ Y_p &= (y_{p1}, y_{p2}, \dots, y_{pm}), \end{aligned} \quad (15)$$

where n is number of input nodes; m is the number of output nodes; l is number of hidden layer nodes; p is number of training samples. This paper n is equal to 2, m is equal to 1, l is equal to 5, and p is equal to 32.

(2) *The Output of the Hidden Layer.* According to the input vector X , the output h of the hidden layer can be calculated, as can be seen in

$$h_{pi} = f\left(\sum_{i=1}^n \omega_{ij} x_{pi} - a_j\right), \quad j = 1, 2, \dots, l, \quad (16)$$

where ω_{ij} is the connection between the input layer and the hidden layer weight and a_j is the threshold of the hidden

layer nodes. $f(x)$ is the activation function of the hidden layer which is shown in

$$f(x) = \frac{1}{1 + e^{-x}}. \quad (17)$$

(3) *The Output of the Output Layer.* Based on output 3 of the hidden layer, the prediction output of BP neural network is calculated, as shown by

$$\hat{y}_{pk} = \sum_{j=1}^l h_{pj} \hat{\omega}_{jk} - b_k, \quad k = 1, 2, \dots, m, \quad (18)$$

where $\hat{\omega}_{jk}$ is the connection weight between the hidden layer and the output layer and b_k is the threshold of output layer nodes.

(4) *Error Calculation.* According to the network forecast output \hat{y}_k and the expected output y_k , then calculate the network prediction error e_k , as shown by

$$e_k = y_{pk} - \hat{y}_{pk}. \quad (19)$$

(5) *Weight Update.* The weight is updated according to the network prediction error (e_k), as shown in

$$\begin{aligned} \omega_{ij} &= \omega_{ij} + \eta h_{pi} (1 - h_{pi}) x(i) \sum_{k=1}^m \hat{\omega}_{jk} e_k \\ \hat{\omega}_{jk} &= \hat{\omega}_{jk} + \eta h_{pj} e_k \\ i &= 1, 2, \dots, n; \quad j = 1, 2, \dots, l; \quad k = 1, 2, \dots, m, \end{aligned} \quad (20)$$

where η is the learning efficiency of BP neural network.

(6) *Threshold Update.* The threshold is updated according to the network prediction error (e_k), as shown in

$$\begin{aligned} a_j &= a_j + \eta h_{pj} (1 - h_{pj}) \sum_{k=1}^m \hat{\omega}_{jk} e_k \\ b_k &= b_k + \eta e_k. \end{aligned} \quad (21)$$

Determine whether the iteration is over; if not, return to step (2). It is noted that all the iterative processes of the BP neural network have a mean square error (mse), as shown in (22). In addition, the average accuracy of the prediction (R) is defined by Kreith et al. (2000) [27] and Wang (2000) [28]. In other words, R represents the goodness of fit, which is used to measure the correlation between prediction output data and training samples data. The closer it is to the value of 1, the better the training network.

$$\text{mse} = \frac{1}{mp} \sum_{p=1}^p \sum_{j=1}^m (\hat{y}_{pj} - y_{pj})^2 \quad (22)$$

$$R = \frac{1}{P} \sum_{i=1}^P R_i = \frac{1}{P} \sum_{i=1}^P \frac{y_{pi}}{\hat{y}_{pi}}, \quad (23)$$

where m is the number of output nodes; p is number of training samples.

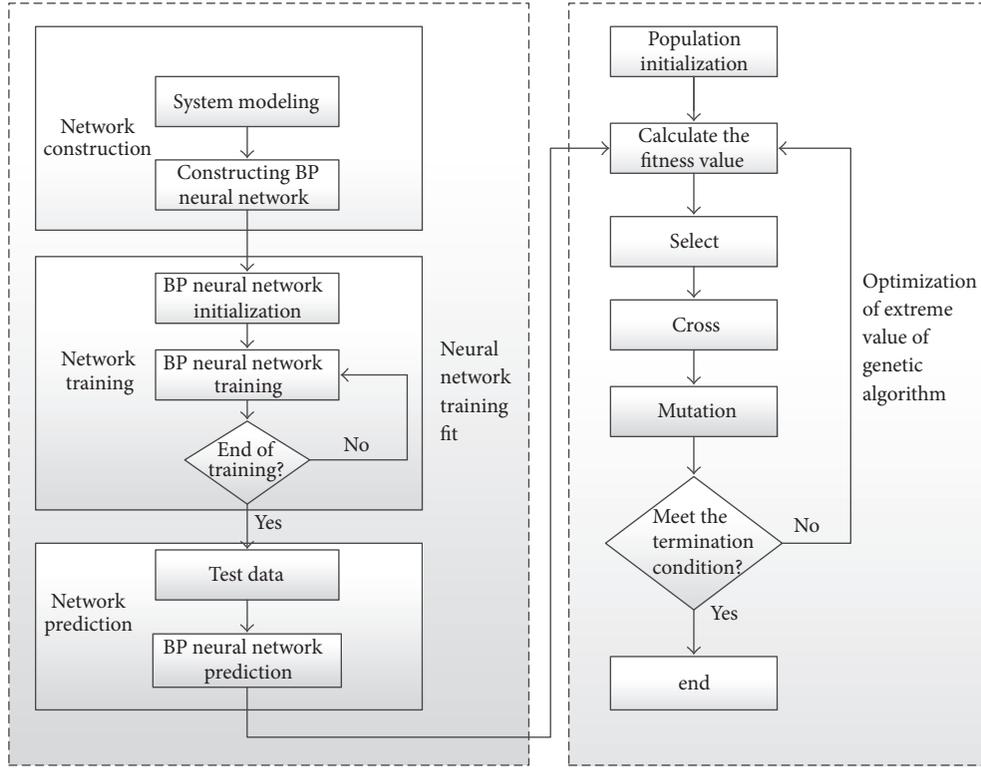


FIGURE 9: Neural network and genetic algorithm flow chart.

5.2. Genetic Algorithm Optimization. Genetic algorithm is a computational model of the biological evolution process of the simulation genetic mechanism. It is a method to search the optimal solution for the feasible solution of the problem. It has strongly global optimization ability. In the genetic algorithm, we will be the individual of each population as a solution to the problem, that is, chromosomes. Genetic algorithm in MATLAB in the operation process is as follows:

- (1) First, we initialize the population and then calculate the fitness value. Finally, we look for the best chromosome from the initial population, which is the solution to the problem.
- (2) Iterative optimization is as follows:
 - (a) Select: first, the solution to the problem is encoded by using the floating-point encoding. This function selects the chromosomes in each generation population for subsequent crossover and mutation. The method used is the roulette selection method.
 - (b) Crossover: this function is a random selection of two chromosomes, according to the crossover probability of determining whether the cross and the cross position are also random.
 - (c) Mutation: this function performs the mutation operation. The mutation chromosomes and mutated positions are randomly selected. Finally, it will check the feasibility of chromosomes; otherwise, it will be recompiled.
 - (d) Result analysis.

After several generations, there is an optimal solution to the problem in the population. The establishment of BP neural network's approximate model and the numerical optimization of the genetic algorithm are shown in Figure 9 [29].

6. Analysis of Vibration Suppression Rate of CFD Simulation Results to the UMP

6.1. Vibration Amplitude Analysis. As is shown in Table 6, this paper has obtained 33 sets of data by CFD simulation. Through these sets of data, we can analyze the influence of dimensionless plate length (L_p/D) and damping (c) on the amplitude of vibration of UMP, as shown in the three-dimensional Figure 10. The x -coordinate of the three-dimensional figure represents the change of the dimensionless plate length (L_p/D), the y -coordinate represents the change of the damping value (c), and the z -coordinate represents the vibration suppression rate (P_y). In addition, it corresponds to the two-dimensional contour line as shown in Figure 11. From these two graphs we can get the effect of the change of dimensionless plate length (L_p/D) and damping value (c) on the vibration suppression rate (P_y). We summarize the following rules.

- (1) On the whole, when the damping value is constant, the effect of the dimensionless plate length (L_p/D) on the UMP is increased first and then decreases with the increase of the dimensionless plate length (L_p/D). When the damping values (c) are at 50, 75, 100, 125, and 175 ($N \times s/m$), respectively, the result of the vibration suppression of the

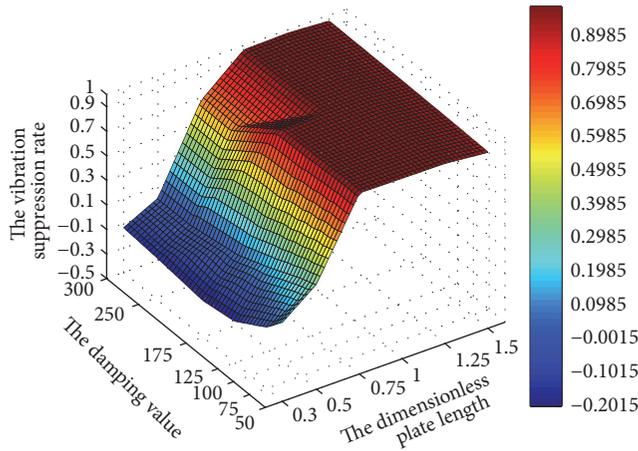


FIGURE 10: Three-dimensional surface cloud diagram of variation of the vibration suppression rate.

UMP is most obvious, reaching more than 0.9 with the change of the dimensionless plate length (L_p/D) from 0.75 to 1.25.

(2) When the dimensionless plate length (L_p/D) is 0.75, the suppression rate (P_y) dropped from 0.955698 to 0.671616, with the damping value (c) changing from 175 to 300 ($N \times s/m$). When the dimensionless plate (L_p/D) is 0.3, the vibration suppression rate (P_y) drops to a negative value, which indicating that it will not suppress the vibration of the UMP in this case. In the dimensionless plate length (L_p/D) at 0.3, 0.5, 0.75, and 1.0, respectively, the vibration suppression rate (P_y) decreases with the increase of the damping value, but it is increasing when the dimensionless plate length (L_p/D) is 1.5. When the dimensionless plate length (L_p/D) is 1.25, the vibration suppression rate (P_y) increases first and then decrease with the increase of the damping value, and the maximum vibration suppression rate (P_y) is 0.979676.

(3) In these data, when $L_p/D = 1.25$, $c = 175$ ($N \times s/m$) is the best data point of the UMP vibration suppression. Its vibration suppression rate (P_y) reached 0.979676.

6.2. Analysis of Lift Coefficient and Drag Coefficient. We study the two different parameters separately and discuss the variation law of the lift coefficient and the drag coefficient of a parameter to the whole UMP, respectively. It should be noted here that the lift coefficient we are studying is the total lift coefficient of whole system. As the drag coefficient of the plate relative to the cylinder drag coefficient is very small, we ignore the plate drag coefficient and the cylinder drag coefficient approximation as the whole system drag coefficient. In order to clearly characterize the force of the whole system, we chose the root mean square (RMS) of the lift coefficient and the averaged drag coefficient in the following study.

(1) *Influence of Dimensionless Plate Length on Lift Coefficient and Drag Coefficient.* Figure 12 shows the variation of the total lift coefficient between different damping values as the dimensionless plate length increases. At the same time, we compare it with the lift coefficient of the bare cylinder.

As can be seen from figure, overall, in the case of the same damping value, the RMS of total lift coefficient decreases first and then increases gradually with the increase of the dimensionless plate length. It can be seen that when the dimensionless plate length is from 0.5 to 0.75, the RMS of total lift coefficient sharply is down from about 0.6 to about 0.15. When the dimensionless plate length is 0.75, 1, 1.25, and 1.5, their RMS of total lift coefficient are less than the bare cylinder. And we can see that the damping value is 175 ($N \times s/m$) and the dimensionless plate length is 1.25 when the RMS of total lift coefficient is the smallest.

As can be seen from Figure 13, all the mean drag coefficient of the cylinder is less than the bare cylinder. The drag coefficient is reduced with the increase of dimensionless plate length. When the dimensionless plate length is 0.75, 1.0, 1.25, and 1.5, the larger the damping value, the smaller the mean drag coefficient, but the opposite when the dimensionless plate length is 0.3 and 0.5.

(2) *Influence of the Damping Value on Lift Coefficient and Drag Coefficient.* Figure 14 is a graph showing a corresponding changes in the RMS of total lift coefficient when the damping value is increased. It is clear from Figure 15 that the RMS of total lift coefficient is much larger than that of the bare cylinder when the dimensionless plate length is 0.3 and 0.5 over the entire damping value change region. The remaining dimensionless plate length is less than the bare cylinder or close to it. In the range where the damping value is greater than 100, the RMS of total lift coefficient of the dimensionless plate length equal to 1.25 is smaller than that of the other dimensionless plates, and when the damping value is 175 ($N \times s/m$), the RMS of total lift coefficient is minimized and the minimum value is 0.048405. The dimensionless plate length 0.75 is relatively variable, and when the damping value is 50 and 75, the RMS of total lift coefficient is the smallest compared to the other dimensionless plate length. When the damping value changes from 175 to 250 ($N \times s/m$), the RMS of total lift coefficient increases dramatically.

In Figure 15, we can see that the mean drag coefficient of the bare cylinder is the largest compared with the all dimensionless plate length when the damping value increases. The mean drag coefficient of the dimensionless plate length of 0.3 and 0.5 increased gradually as the damping value increases. However, the remaining dimensionless plate length is reduced. The longer the dimensionless plate length is, the smaller the mean drag coefficient in the region of the global damping range will be.

Through the above analysis, we can get the following conclusions.

(1) When the RMS of total lift coefficient of the entire system is less than the RMS of lift coefficient of the bare cylinder, the vibration suppression effect is obvious, and the vibration suppression rate is more than 0.9.

(2) The mean drag coefficient of the cylinder with the plate is smaller than the bare cylinder. That is to say, the plate can reduce the drag force.

6.3. Analysis of Wake Structure. Through the numerical simulation results, when the dimensionless plate length is 1.25

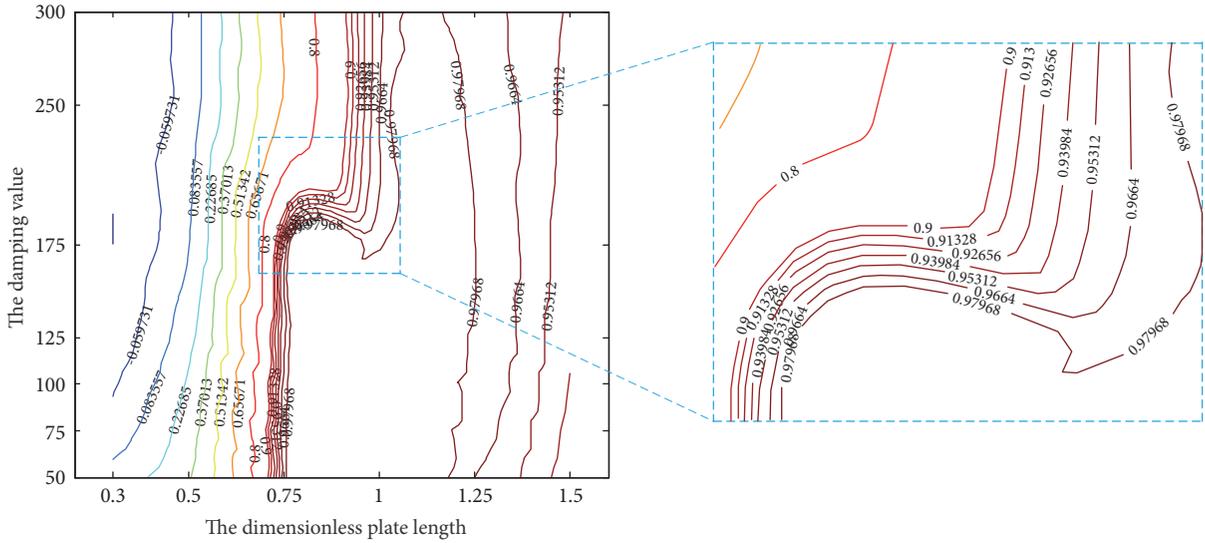


FIGURE 11: Two-dimensional contour line of variation of the vibration suppression rate.

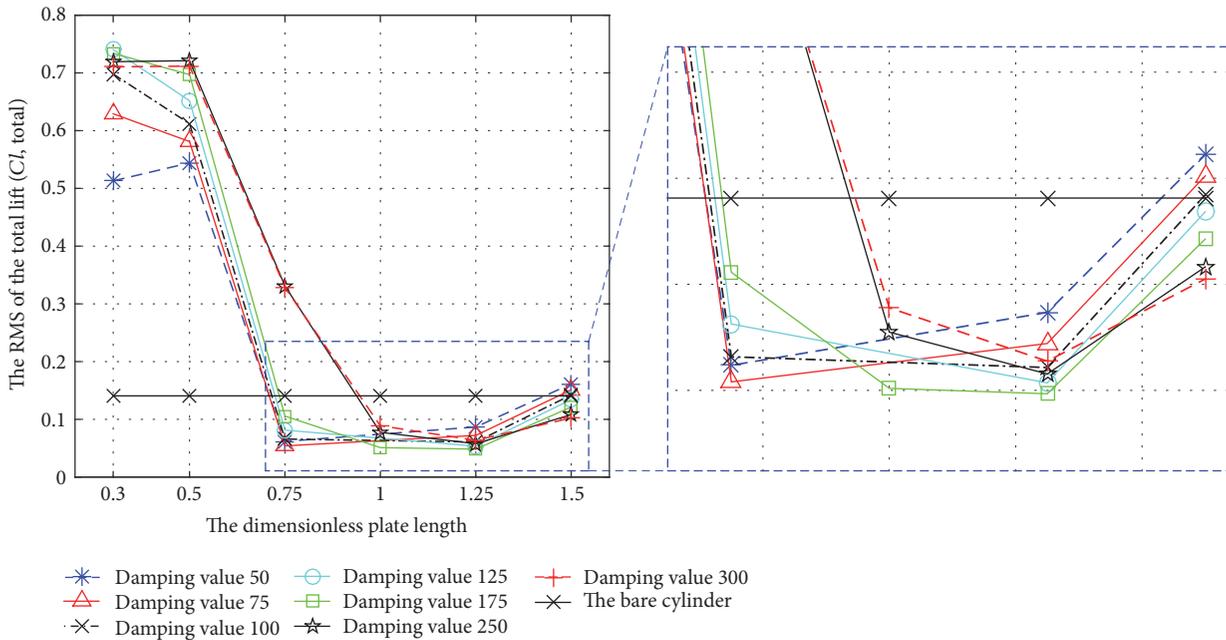


FIGURE 12: The variation of the total lift coefficient (RMS) with the changing of the dimensionless plate length.

and the damping value is 175, the plate has the best effect of suppressing the vibration of the UMP. So we chose the case to analyze its vortex pattern and vortex shedding. Vortex shedding patterns can refer to the literature of Williamson and Roshko (1988) [30], and there are 2S, 2P, and 2T.

As can be seen from Figure 16, the graph is a contours of vorticity with a dimensionless plate length of 1.25 and a damping value of 175 ($N \times s/m$). The contour of vorticity shows that its structure is a typical 2S vortex pattern. This shows that the plate for the UMP wake region has a certain impact. The shear layer is then extended to the tip of the plate.

Figure 17 is a description of the interval of $(1/4)T$ cycles and then draws the different interval cycle contours of vorticity and pressure. As can be seen from the figure, the plate prevents the interaction between the shear layers. The vortex shedding extends from the surface of the cylinder to the wake end of the plate. So that the pulsating force on the cylindrical surface is weakened. As can be seen from the vorticity cloud, the vortex is moving from the root of the plate to the tip. On the sides of the plate there is always only one vortex, one vortex is at the same time as the tip of the plate, and the other vortex is formed at the root of the plate. From

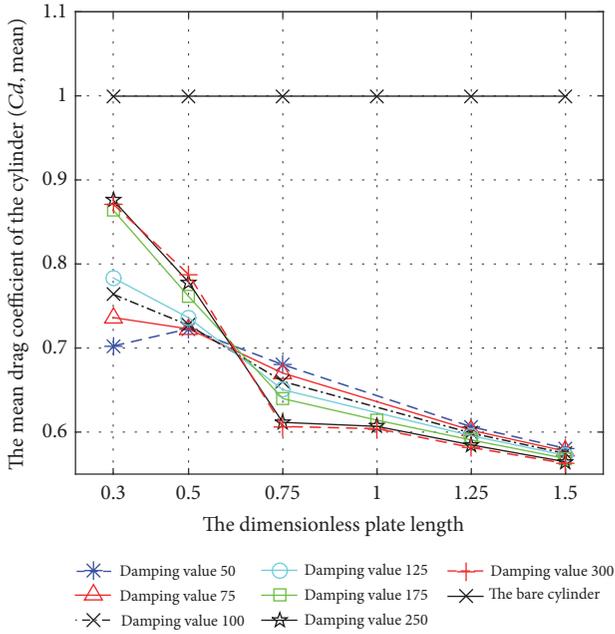


FIGURE 13: The variation of the drag coefficient (mean).

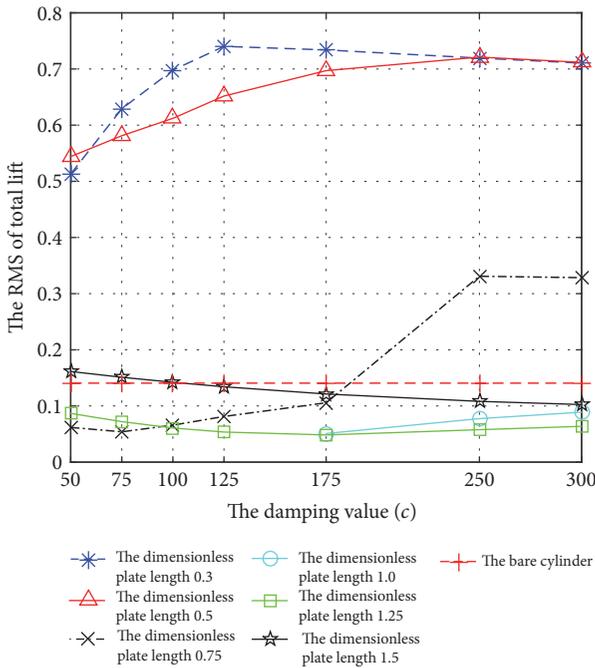


FIGURE 14: The variation of the RMS of total lift coefficient with the changing of the damping value.

the pressure cloud shows that the vortex of the cylindrical surface can not fall off the rear surface of the cylinder for the presence of the partition plate but is extended to the tip of the partition plate. At this time from the figure can be seen, the cylindrical surface of the back pressure on both sides is difficult to change. It was changed significantly at the end of the board, and the pressure on the surface of the cylinder was

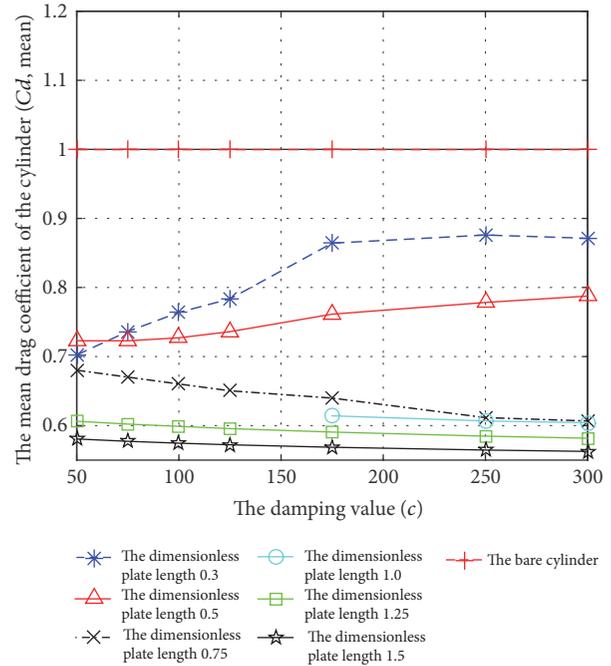


FIGURE 15: The variation of the mean drag coefficient with the changing of the damping value.

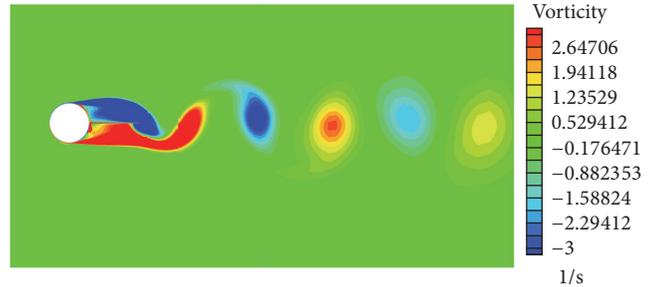


FIGURE 16: A contour of vorticity with a dimensionless plate length of 1.25 and a damping value of 175.

observed to be opposite to the pressure of the plate. This helps to reduce lift.

7. Neural Network and Genetic Algorithm Optimization

7.1. Neural Network Training

(1) *Selection of Sample Data.* In this study, we use MATLAB software as a neural network and genetic algorithm optimization platform. First, we have to select the reasonable sample data points. As can be seen from Table 6, the suppression rate of the third to eight data is negative. This situation does not achieve the effect of vibration suppression; it can not be used as the optimization of data points. The first group is bare cylindrical data, so it can not be used as an optimized

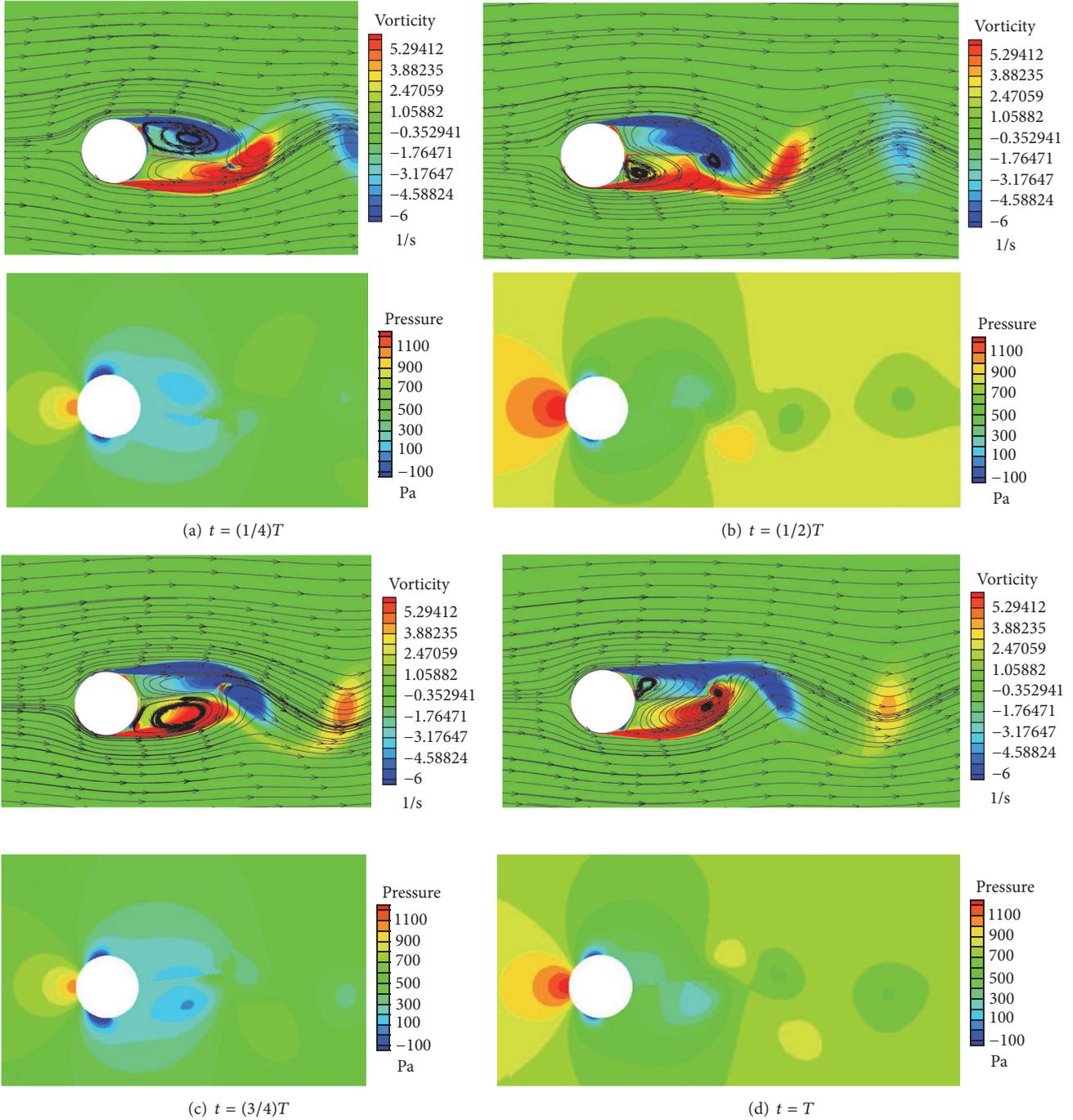


FIGURE 17: The different interval cycle contours of vorticity and pressure.

sample point. Among them, we use random 19 sets of data for training data, the remaining 13 sets of data as test data.

(2) *Network Model Establishment.* From Figure 7 we can see that we use 3-layer neural network to process 32 sets of data, and the number of hidden layer take 5.

(3) *Network Training.* We set the target error of the network (`net.trainParam.goal`) to 0.000004, the learning rate

(`net.trainParam.lr`) to 0.1, and the number of training steps (`net.trainParam.epochs`) to 100. At the same time, we set the mean squared errors (`mse`) as a performance function.

According to the above settings, the training of BP neural network is carried out until the training network meets the intended target. As a result, the prediction of the BP neural network is shown in Figure 18. As can be seen from the figure, the training network is very close to the data. Table 7 shows the comparison of the neural network prediction output with

TABLE 7: Comparison of the neural network prediction output with the test sample data points.

Number	Test sample data	Prediction data	Absolute error	Relative error (%)
(1)	0.949309	0.9415	0.007809093	0.822608
(2)	0.009622	0.01069	0.001067986	11.099397
(3)	0.978471	0.9766	0.001870826	0.191199
(4)	0.971677	0.8949	0.076777254	7.901518
(5)	0.358304	0.3872	0.028896097	8.064689
(6)	0.249475	0.2568	0.007325429	2.936343
(7)	0.955698	0.7909	0.164798081	17.243739
(8)	0.977124	0.931100	0.046023748	4.710125
(9)	0.038191	0.065120	0.026928866	70.510779
(10)	0.965197	0.861200	0.10399661	10.774656
(11)	0.006293	-0.008060	0.014353191	228.074937
(12)	0.673212	0.579600	0.093611759	13.905247
(13)	0.966174	0.970000	0.003826088	0.396004

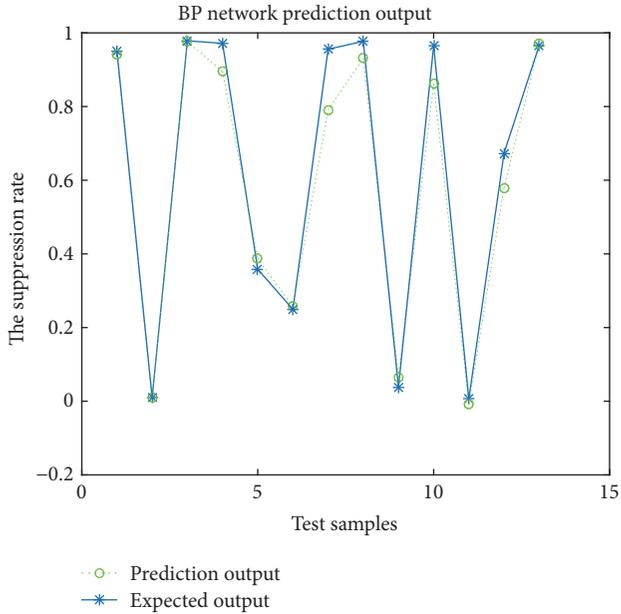


FIGURE 18: BP neural network prediction results.

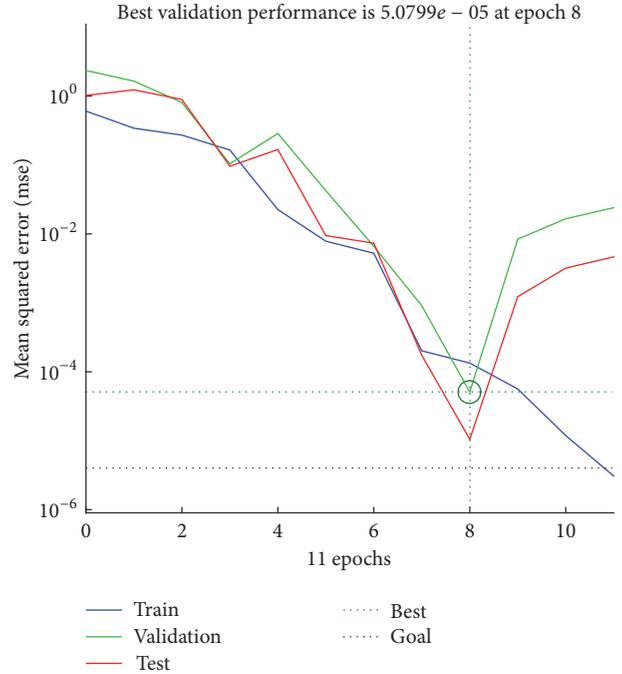


FIGURE 19: The performance of the neural network.

the test sample data points. The maximum absolute error between them is 0.164798081 and the minimum relative error is 0.191199 percent. We can get the relative error in the range of 0.191199 to 228.074937 percent. What needs to be explained here is that we can see that the predicted output data is negative, indicating that the condition does not play the role of vibration suppression but contributed to the vibration. So it is possible that a negative suppression vibration rate with the lower vibration suppression rate in the sample data appears. So its relative error is very large, reaching 228.074937 percent.

Figure 19 shows the performance of the neural network. This figure shows that the Validation Performance is $5.0799e-5$ at epoch 8, and it has been very close to our target error of the network. Figure 20 shows the regression analysis of the BP

neural network. As can be seen from the figure, the training set, the validation set, and the test set of complex correlation number (R) are very close to 1. It is shown that the constructed BP neural network has high prediction accuracy.

7.2. Genetic Algorithm Optimization. Then we use the genetic algorithm to optimize the previously trained BP neural network. The suppression rate of the UMP is taken as the fitness. The fitness curve calculated by the genetic algorithm is shown in Figure 21

As can be seen from Figure 21, when the iteration to the beginning is very fast convergence, the optimal fitness value (P_y) tends to be stable, and its value is 0.9878. At this

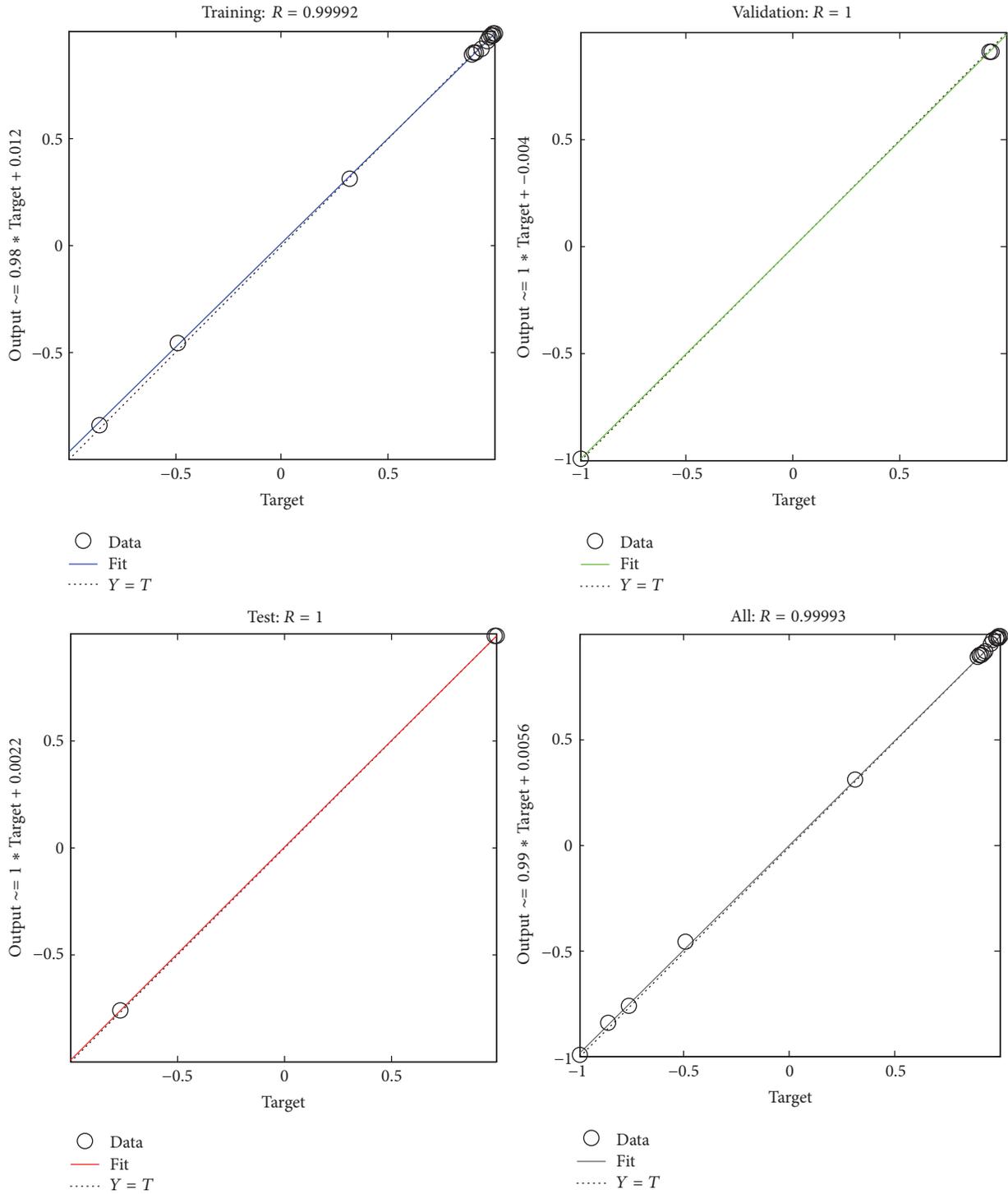


FIGURE 20: The regression analysis of the BP neural network.

time we can get an optimized optimal vibration suppression rate (P_y) as well as the corresponding dimensionless plate (L_p/D) length and damping value (c). They are $P_y = 0.9878$, $L_p/D = 1.0342$, and $c = 57.9631$ ($N \times s/m$), respectively. The comparison data are shown in the Table 8. Optimization before and after the data comparison shows that the suppression rate increased by 0.82925.

8. CFD Validation

In order to verify the accuracy and validity of BP neural network and genetic algorithm optimization results, we have obtained the optimization results and verified the optimized results with FLUENT 15.0 numerical test method. At the same time, we must guarantee the same simulation conditions as

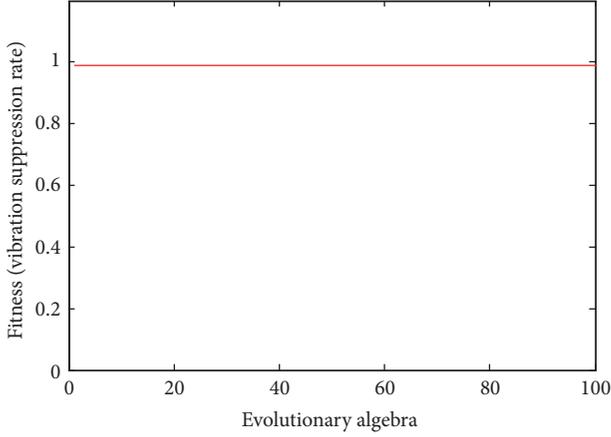


FIGURE 21: The fitness curve calculated by the genetic algorithm.

TABLE 8: Comparison of data before and after optimization.

Parameter	Before optimization	After optimization
L_p/D	1.25	1.0342
c ($N \times s/m$)	175	57.9631
P_y	0.979676	0.9878

TABLE 9: Comparison of the results of the optimized data and CFD simulation.

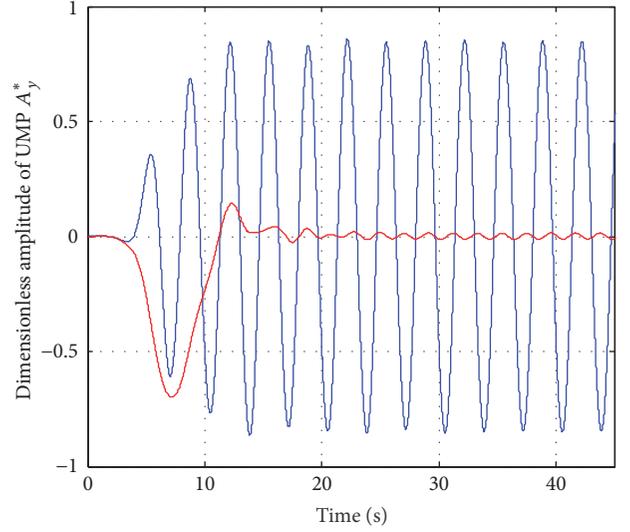
Optimization results	CFD simulation results	Absolut error
P_y	0.9878	0.0037

this article to carry out CFD simulation. The final simulation results are shown in Table 9.

As can be seen from Table 9, the results of BP neural network and generic algorithm optimization are closed to those of CFD simulation, and the absolute error is only 0.0037. It is shown that the BP neural network and generic algorithm are effective. We also found that the results of the vibration suppression rate optimized by the BP neural network and generic algorithm ($P_y = 0.9878$) are larger than $P_y = 0.979676$ of the samples we use, reaching the highest suppression vibration rate. Similarly, the vibration suppression rate is $P_y = 0.9841$ under the CFD verification with the optimal parameters ($L_p/D = 1.0342$, $c=57.9631$ ($N \times s/m$)), which is also the maximum of the sample data. This shows that the BP neural network and generic algorithm can be based on our objective function (vibration suppression rate) to optimize the parameters to achieve the optimal value.

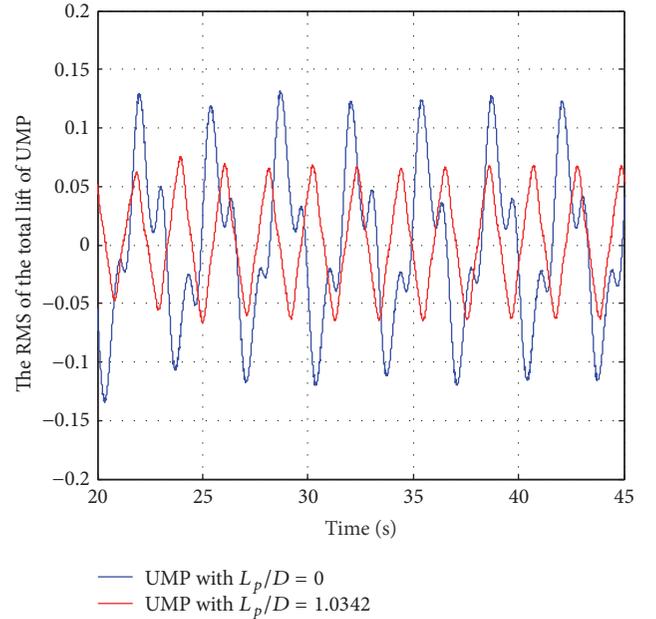
8.1. Comparing the Result of Bare UMP and CFD Simulation Based on Optimal Parameters. We compare the results of the analysis of bare UMP and CFD simulation based on optimal parameters, mainly for dimensionless amplitude, wake structure, and total lift coefficient.

(1) *Dimensionless Amplitude.* As shown in Figure 22, it is clear that the dimensionless amplitude of the UMP with plate ($L_p/D = 1.0342$) is much smaller than that of the bare UMP.



— UMP with $L_p/D = 0$
 — UMP with $L_p/D = 1.0342$

FIGURE 22: Dimensionless amplitude of the bare UMP and UMP with plate based on optimal parameter.



— UMP with $L_p/D = 0$
 — UMP with $L_p/D = 1.0342$

FIGURE 23: The RMS of total lift coefficient of the UMP with plate and the bare UMP.

The dimensionless amplitude s of the UMP with plate and the bare UMP are 0.013508 and 0.847339, respectively. The suppression rate of the CFD simulation based on optimal parameters is 0.9841.

(2) *The RMS of the Total Lift.* As shown in Figure 23, the RMS of total lift coefficient is also reduced relative to the bare UMP by comparing the CFD simulation results with the optimization results (obtained from the BP neural network

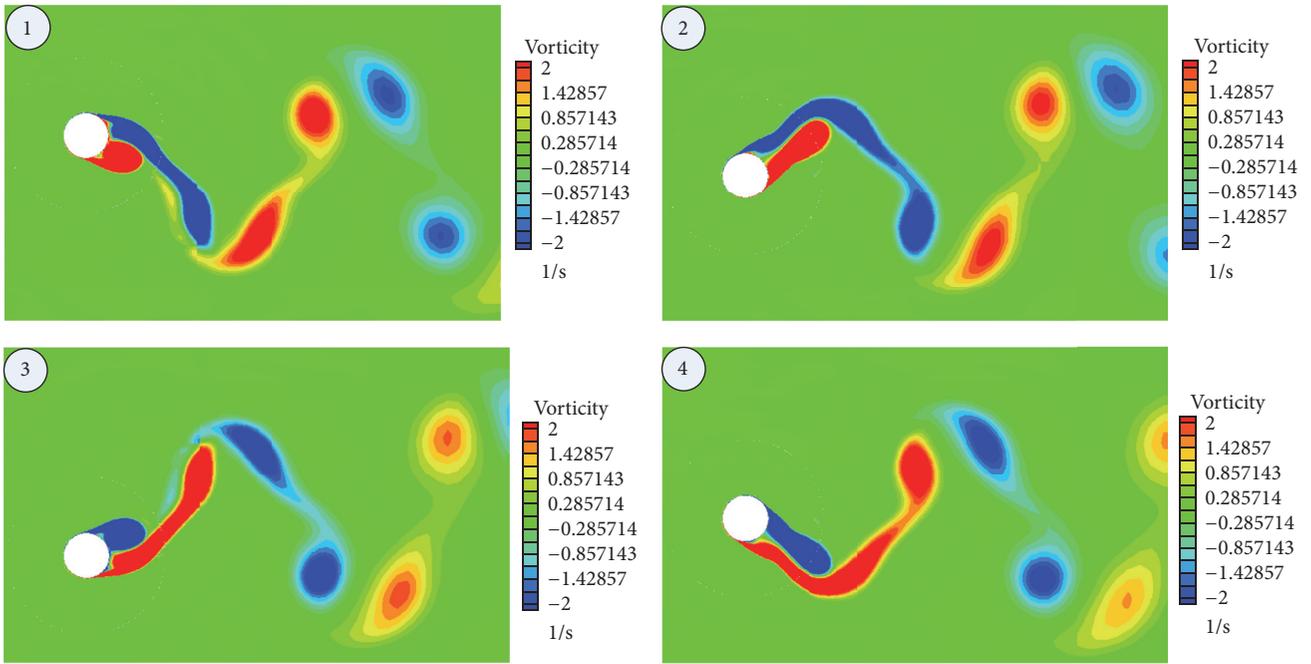
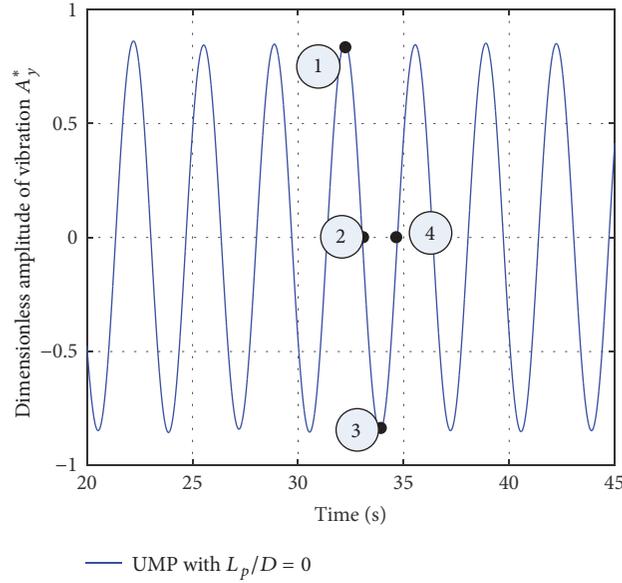


FIGURE 24: The vorticity cloud images of bare UMP.

and genetic algorithm). The RMS of total lift coefficient of the UMP with plate and the bare UMP are 0.040929 and 0.140653, respectively.

(3) *Wake Structure.* As shown in Figures 24 and 25, they are the vorticity cloud images of bare UMP and UPM with $L_p/D = 1.0342$, respectively. We only selected the four points to show vorticity cloud images. It was observed that the plate changed the wake structure pattern of the UMP, and the wake structure pattern changed from the 2P mode of the bare UMP to the 2S mode of the UMP with the plate. The interaction of the shear layer is prevented by the plate. The plate extends the

vortex shedding, as the vortex is removed from the UMP and reattached to the plate.

Figures 26(a) and 26(b) are a pressure cloud with the same position as the above vortex cloud. The overall can be seen in Figure 26(a), and the bare UMP's wake region pressure changes significantly. At this point, the front portion of the bare UMP by the pressure distribution is not uniform. So it produced a relatively large lift. On the contrary, in Figure 26(b), the pressure on both sides of the plate behind the UMP has not changed greatly, because the vortex is extended to the tip of the plate to shed in the wake region of the UMP with the plate.

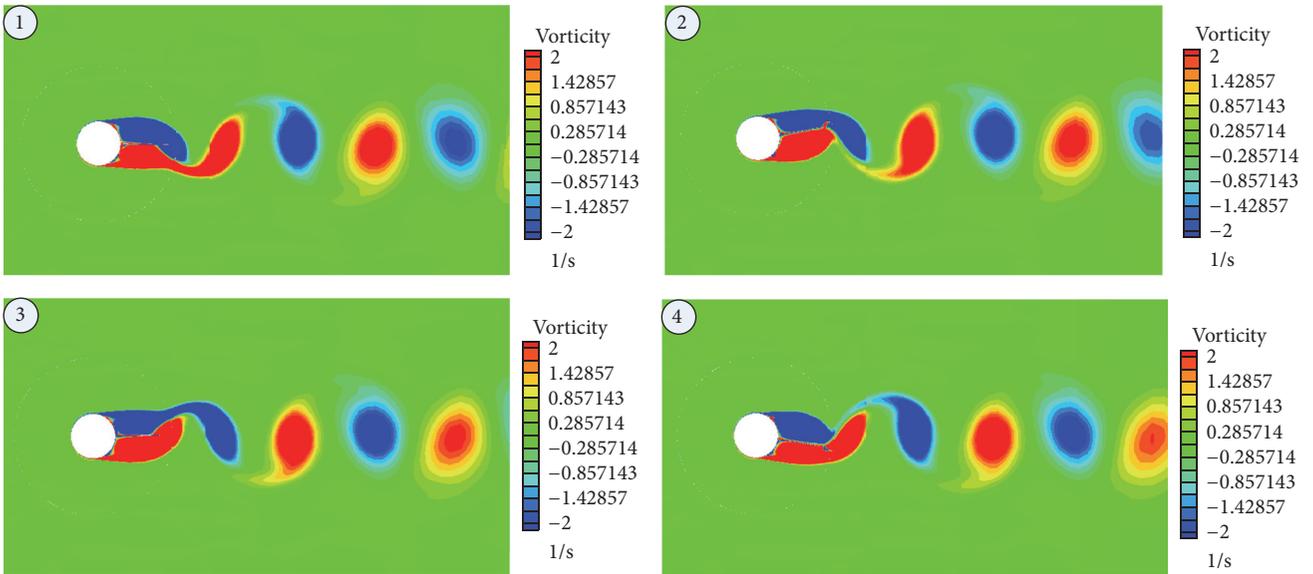
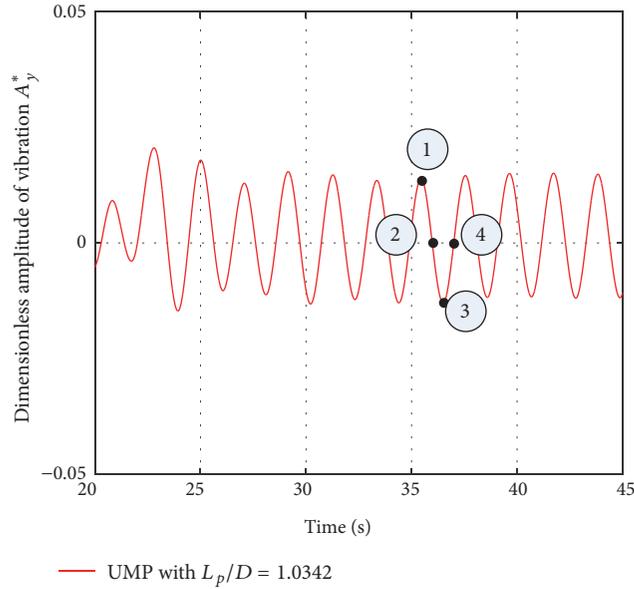


FIGURE 25: The vorticity cloud images of UPM with $L_p/D = 1.0342$.

9. Conclusion

Under the action of current flow, the UMP suspended in seawater will undergo vortex-induced vibration, so that it can lose its value. In this paper, we studied the effect of splitter plate on the vibration of the UMP. The main parameters of the plate are the dimensionless plate length (L_p/D) and the damping value (c) between the UMP and plate. For us to use CFD numerical simulation of the UMP vibration, it will increase the calculation cost and time with the more situations. So, we consider using BP neural network and generic algorithm to optimize these two parameters in order to minimize the vibration of the UMP. For this study, the following conclusions can be drawn.

(1) We selected 38 sample data points for CFD simulation and extracted the results. From the finite discrete

parameter data point, we can see that the amplitude of the UMP is the smallest when the dimensionless plate length (L_p/D) is 1.25, the damping (c) is 175 ($N \times s/m$), and the maximum suppression rate (P_y) is 0.979676. The effect on the suppression vibration achieves the best results. When the dimensionless plate length is 0.3 and 0.5, regardless of the fact that the damping value is how much, its vibration suppression effect is poor and may increase the vibration amplitude. The vibration suppression effect is better in the dimensionless plate length of 0.75 to 1.5, and the best damping ranges vary with dimensionless plate length.

(2) We use MATLAB software as a neural network and genetic algorithm optimization platform. First, we selected the appropriate 32 sets of data as an optimized variable and the suppression rate as the objective function for BP neural network and genetic algorithm. The optimization results

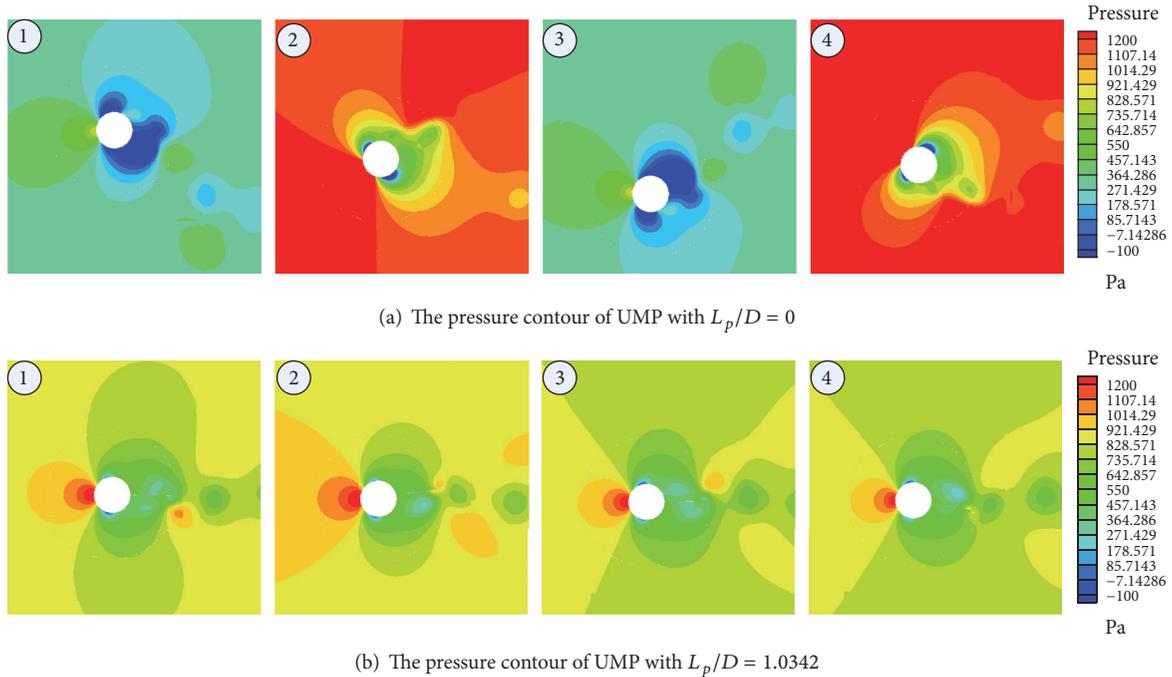


FIGURE 26: Pressure cloud of the bare UMP and UMP with plate.

show that the optimum suppression rate is 0.9878, and the corresponding dimensionless plate length (L_p/D) is 1.0342 and the damping value (c) is 57.9631 ($N \times s/m$).

(3) In order to verify the accuracy of the optimization, we perform the CFD numerical simulation with the optimized parameters and compare the theoretical optimization results with the CFD simulation result. As can be seen from Table 9, the results of BP neural network and generic algorithm optimization are closed to those of CFD simulation, and the absolute error is only 0.0037. It is shown that the BP neural network and generic algorithm are effective.

Through the study of this paper, we obtained the parameter value when the vibration suppression effect is the best by BP neural network and genetic algorithm and CFD validation.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Science Foundation of China (Grants nos. 51179159 and 61572404) and the Shaanxi Province Youth Science and Technology New Star Project (Grant no. 2016KJXX-57).

References

- [1] L. Ding, L. Zhang, Z.-Q. Yang et al., "Effect of splitter plate on vortex-induced vibration of circular cylinder at high Reynolds number," *Chinese Journal of Mechanical Engineering*, vol. 49, no. 14, pp. 133–139, 2013.
- [2] B. Tan, J.-S. Wang, F. Gu et al., "Numerical simulation of vortex-induced vibration of riser using separation disk," *Hydrochemistry Research and Progress*, vol. 24, no. 1, pp. 43–48, 2009.
- [3] J. Wang, H. Liu, F. Gu, and P. Zhao, "Numerical simulation of flow control on marine riser with attached splitter plate," in *Proceedings of the ASME 2010 29th International Conference on Ocean, Offshore and Arctic Engineering*, pp. 489–498, June 2010.
- [4] M. Amiraslarpour, J. Ghazanfarian, and S. E. Razavi, "Drag suppression for 2D oscillating cylinder with various arrangement of splitters at $Re = 100$: a high-amplitude study with OpenFOAM," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 164, pp. 128–137, 2017.
- [5] Y. Qiu, Y. Sun, Y. Wu, and Y. Tamura, "Effects of splitter plates and Reynolds number on the aerodynamic loads acting on a circular cylinder," *Journal of Wind Engineering & Industrial Aerodynamics*, vol. 127, pp. 40–50, 2014.
- [6] M. Lou, Z. Chen, and P. Chen, "Experimental investigation of the suppression of vortex induced vibration of two interfering risers with splitter plates," *Journal of Natural Gas Science and Engineering*, vol. 35, pp. 736–752, 2016.
- [7] G. R. S. Assi, P. W. Bearman, and N. Kitney, "Low drag solutions for suppressing vortex-induced vibration of circular cylinders," *Journal of Fluids and Structures*, vol. 25, no. 4, pp. 666–675, 2009.
- [8] G. R. S. Assi, P. W. Bearman, and M. A. Tognarelli, "On the stability of a free-to-rotate short-tail fairing and a splitter plate as suppressors of vortex-induced vibration," *Ocean Engineering*, vol. 92, pp. 234–244, 2014.
- [9] B. Gozmen, H. Akilli, and B. Sahin, "Passive control of circular cylinder wake in shallow flow," *Measurement*, vol. 46, no. 3, pp. 1125–1136, 2013.
- [10] H. Zhu, Y. Lin, Q. Jia, and X. Yang, "Simulations of suppressive effect of VIV on marine riser with splitter plates," in *Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology (IACSIT '10)*, vol. 6, p. 5, IEEE

- China Council, IEEE Beijing Section Sichuan Computer Federation, 2010.
- [11] F. Gu, J. S. Wang, X. Q. Qiao, and Z. Huang, "Pressure distribution, fluctuating forces and vortex shedding behavior of circular cylinder with rotatable splitter plates," *Journal of Fluids and Structures*, vol. 28, pp. 263–278, 2012.
- [12] H. Akilli, B. Sahin, and N. F. Tumen, "Suppression of vortex shedding of circular cylinder in shallow water by a splitter plate," *Flow Measurement and Instrumentation*, vol. 16, no. 4, pp. 211–219, 2005.
- [13] Y. Z. Law and R. K. Jaiman, "Wake stabilization mechanism of low-drag suppression devices for vortex-induced vibration," *Journal of Fluids and Structures*, vol. 70, pp. 428–449, 2017.
- [14] G. R. S. Assi and P. W. Bearman, "Transverse galloping of circular cylinders fitted with solid and slotted splitter plates," *Journal of Fluids and Structures*, vol. 54, pp. 263–280, 2015.
- [15] L. Ma, S. Hu, M. Qiu, Q. Li, and Z. Ji, "Energy consumption optimization of high sulfur natural gas purification plant based on back propagation neural network and genetic algorithms," *Energy Procedia*, vol. 105, pp. 5166–5171, 2017.
- [16] W.-Z. Jiang, S.-Q. Duan, C.-M. Yang et al., "Optimal design of centrifugal pump impeller based on CFD simulations and BP neural network," *Machine Tool & Hydraulic*, vol. 44, no. 22, pp. 67–70, 2016.
- [17] C.-J. Liang, G.-L. Yang, X.-F. Wang et al., "Dynamic optimization of artillery structure based on neural network and genetic algorithm," *Journal of Ordnance Industry*, vol. 36, no. 5, pp. 789–794, 2015.
- [18] A. M. C. Smith, C. Yang, H. Ma, P. Culverhouse, A. Cangelosi, and E. Burdet, "Novel hybrid adaptive controller for manipulation in complex perturbation environments," *PLoS ONE*, vol. 10, no. 6, Article ID e0129281, 2015.
- [19] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [20] H. Safikhani, A. Abbassi, A. Khalkhali, and M. Kalteh, "Multi-objective optimization of nanofluid flow in flat tubes using CFD, Artificial Neural Networks and genetic algorithms," *Advanced Powder Technology*, vol. 25, no. 5, pp. 1608–1617, 2014.
- [21] H. Avci, D. Kumlutaş, Ö. Özer, and M. Özşen, "Optimisation of the design parameters of a domestic refrigerator using CFD and artificial neural networks," *International Journal of Refrigeration*, vol. 67, pp. 227–238, 2016.
- [22] W. Tian, Z. Mao, and F. Zhao, "Design and numerical simulations of a flow induced vibration energy converter for underwater mooring platforms," *Energies*, vol. 10, no. 9, p. 1427, 2017.
- [23] W. Tian, Z. Mao, X. An, B. Zhang, and H. Wen, "Numerical study of energy recovery from the wakes of moving vehicles on highways by using a vertical axis wind turbine," *Energy*, vol. 141, pp. 715–728, 2017.
- [24] ANSYS Inc, *Fluent 15.0 User's Guide*, 2016.
- [25] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, pp. 2004–2016, 2014.
- [26] X. Wang, C. Yang, Z. Ju et al., "Robot manipulator self-identification for surrounding obstacle detection," *Multimedia Tools & Applications*, vol. 76, pp. 6495–6520, 2017.
- [27] E. F. Kreith, M. J. Moran, and G. Tsatsaronis, "Engineering thermodynamics," in *The CRC Handbook of Thermal Engineering*, F. Kreith, Ed., CRC Press LLC, Boca Raton, Fla, USA, 2000.
- [28] B. Wang, *Heat Transfer Science and Technology 2000*, Higher Education Press, 2000.
- [29] X.-C. Wang, *There Are 43 Case Studies of MATLAB Neural Network*, Beijing University of Aeronautics and Astronautics Press, 2013.
- [30] C. H. K. Williamson and A. Roshko, "Vortex formation in the wake of an oscillating cylinder," *Journal of Fluids and Structures*, vol. 2, no. 4, pp. 355–381, 1988.

Research Article

Layout Optimization of Two Autonomous Underwater Vehicles for Drag Reduction with a Combined CFD and Neural Network Method

Wenlong Tian,^{1,2} Zhaoyong Mao,^{1,2} Fuliang Zhao,¹ and Zhicao Zhao³

¹School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China

²Key Laboratory for Unmanned Underwater Vehicle, Northwestern Polytechnical University, Xi'an 710072, China

³Xi'an Institute of Applied Optics, Xian 710065, China

Correspondence should be addressed to Wenlong Tian; tianwenlong@nwpu.edu.cn

Received 8 August 2017; Accepted 12 November 2017; Published 7 December 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Wenlong Tian et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents an optimization method for the design of the layout of an autonomous underwater vehicles (AUV) fleet to minimize the drag force. The layout of the AUV fleet is defined by two nondimensional parameters. Firstly, three-dimensional computational fluid dynamics (CFD) simulations are performed on the fleets with different layout parameters and detailed information on the hydrodynamic forces and flow structures around the AUVs is obtained. Then, based on the CFD data, a back-propagation neural network (BPNN) method is used to describe the relationship between the layout parameters and the drag of the fleet. Finally, a genetic algorithm (GA) is chosen to obtain the optimal layout parameters which correspond to the minimum drag. The optimization results show that (1) the total drag of the AUV fleet can be reduced by 12% when the follower AUV is located directly behind the leader AUV and (2) the drag of the follower AUV can be reduced by 66% when it is by the side of the leader AUV.

1. Introduction

Autonomous underwater vehicles (AUVs) are a kind of self-sailing, self-executing underwater robots, which play an important role in expanding people's knowledge of the ocean. AUVs are capable of a wide range of applications, such as pipeline inspection [1, 2], underwater search and rescue, mine-sweeping [3], and oceanographic exploration [4]. The tasks of current AUVs are relatively single, and in the future, AUVs will be able to perform a variety of tasks. Therefore, AUVs will be composed of more complex structures and control systems, as well as more sensors and functional modules. Another solution for complex tasks is to use a fleet of simple AUVs, among which each one has a specific function, so that the fleet can complete complex tasks with simple AUVs and at a lower cost.

When the AUVs travel in a fleet, the layout of the fleet becomes a key problem. Observations of animal motions such as birds in flocks suggest that some energy benefit may

be obtained by certain fleet shapes [5]. It is known to all that migrating geese fly in a V or I formation for long distance migrations (Figure 1). The leader goose produces high speed updrafts in wake, which help the follower geese to save energy. Similarly, the leader AUV in the fleet produces low speed wake and low pressure side flows. Therefore, the follower AUVs could possibly use the velocity difference or pressure difference to minimize the drag and save energy.

The drag of an individual AUV can be reduced by optimizing the shape of the hull, propeller, and surface control [6, 7]. Alvarez et al. introduced an optimization method for the shape optimization of an AUV moving near the water surface [8]. They used a panel model to predict the wave resistance of a revolution body and a simulated annealing algorithm to optimize the geometric parameters that minimize the wave resistance. Kim et al. proposed a Computational Fluid Dynamics (CFD) method for the optimization of monohull ships with the minimum drag [9]. Joung et al. proposed a procedure using the CFD method

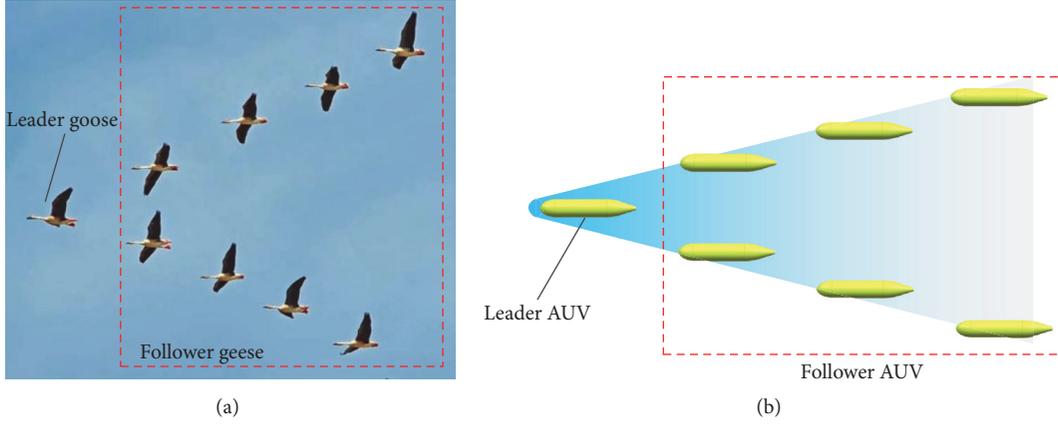


FIGURE 1: (a) Migrating geese and (b) possible layout of an AUV fleet.

and a Design of Experiments (DOE) optimization for the hull optimization [10]. Sun et al. established an energy consumption model for an underwater glider and optimized the shape of the glider with a combined CFD and Efficient Global Optimization (EGO) method [11].

CFD simulations can predict the hydrodynamic performance of the AUV with high accuracy, however, at the cost of time. Combined optimization method based on the CFD, Back-Propagation Neural Network (BPNN), and Genetic Algorithm (GA) can efficiently save the simulation time to obtain the optimal target. BPNN and GA are used at some of the nonlinear data to find the optimal value [12, 13]. Safikhani et al. studied the multiobjective optimization of nanofluid flow in flat tubes, which combined with a CFD, artificial neural networks, and genetic algorithms [14]. They got important design information about nanofluids and flat tubes. Avci et al. studied the optimization of the design parameters of a home refrigerator using CFD and artificial neural network [15]; they showed that CFD simulation and the ANN can determine the best value for the refrigerator design.

Inspired by the above research, this paper introduces an optimization procedure of an AUV fleet, which contains two AUVs, to minimize its drag. The layout of the fleet is defined by two nondimensional parameters. Three-dimensional CFD simulations are performed to find the drag of each case. Then BPNN and GA optimization method is used to find the optimal layout which has the minimum drag.

2. Geometry Configuration

2.1. The AUV Hull Design. The AUV considered in this paper has a torpedo-like axisymmetric shape, which is characterized with a length of $L = 1.3$ m and a maximum diameter of $D = 0.2$ m, resulting in a slenderness ratio (the ratio of the length to the diameter) of 6.5. The two-dimensional sketch of the AUV is shown in Figure 2. The designed AUV is composed of a nose section, a middle section, and a tail section. The lengths of the three sections are 0.2 m, 0.7 m, and 0.4 m, respectively. In order to obtain a low-drag shape, the Myring Equation [17] is used to design the shape of the AUV nose. The Myring Equation is widely used in the

TABLE 1: Dimensional parameters of the AUV.

D (m)	D_t (m)	L (m)	L_h (m)	L_c (m)	L_t (m)	θ ($^\circ$)
0.20	0.04	1.30	0.20	0.70	0.40	30

design of AUV hulls [10]. The expression for the nose shape is determined by

$$r(x) = \frac{1}{2}D \left[1 - \left(\frac{x - L_h}{x} \right)^2 \right]^{1/1.8}, \quad (1)$$

where x is the position along the rotation axis, r is the radius at a specific x , and L_h is the length of the nose. The tail section is smoothly transitioned from the mid-section to a circular cone, which has a cone angle of 30° . The diameter of the rear surface is 0.04 m. Table 1 shows the values of the dimensional parameters.

2.2. Parameters of the AUV Fleet. As shown in Figure 3, a coordinate frame is assigned to the nose of the leader AUV, with its horizontal axis pointing right and its vertical axis pointing to the side of the follower AUV. The layout of the AUV fleet can be characterized by two parameters, a and b , which represent the relative location of two adjacent AUVs in the fleet. To eliminate the dimensional effects, a and b are normalized by dividing the length of the AUV,

$$a = \frac{y_f}{L}, \quad (2)$$

$$b = \frac{x_f}{L},$$

where x_f and y_f are the position of the follower AUV in the proposed coordinate frame.

A series of fleet layouts can be described by changing the two design parameters, a and b . Before the optimization procedure, 53 groups of CFD simulations are performed as sample points with a varying from 0.00 to 0.60 and b varying from 0.00 to 2.00. It should be noted that simulations for $a \leq 0.2$ and $b \leq 1.00$ are not performed because of the geometric interference between the two AUVs in these situations.

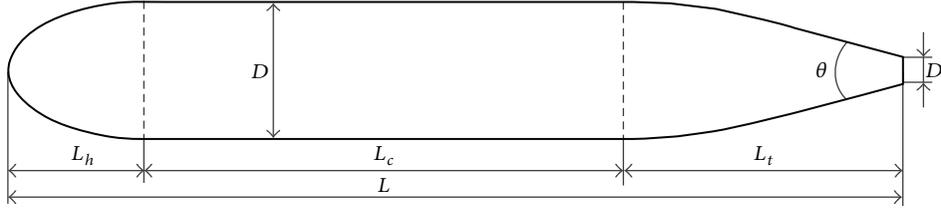


FIGURE 2: Shape of the AUV.

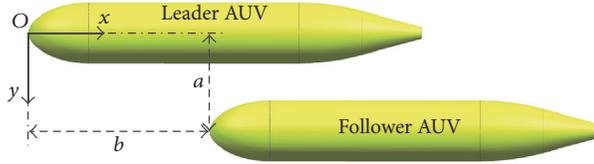


FIGURE 3: Parameter definition for the AUV fleet.

This study aims optimize the drag of the AUV fleet. The drag is normalized in the following way:

$$\begin{aligned} C_{dl} &= \frac{F_{xl}}{(1/2)\rho SU^2}, \\ C_{df} &= \frac{F_{xf}}{(1/2)\rho SU^2}, \\ C_{dsum} &= C_{dl} + C_{df}, \end{aligned} \quad (3)$$

where C_{dl} , C_{df} , and C_{dsum} are the drag coefficient of the leader AUV, the follower AUV, and the fleet, respectively, ρ is the water density and is equal to 998 kg/m^3 , S is the area of the cross-section of the AUV, and U is the velocity of the AUV.

3. CFD Method

The drag coefficients and the flow structures for each simulation case are obtained by the CFD approaches separately. A group of transient representations of the three-dimensional flow are carried out to improve the efficiency of the optimization process. Further, the results of this numerical model will be compared with existing experimental data for the model validation, which will be discussed in Section 3.4.

3.1. Governing Equations. The prediction of the flow around the AUVs is based on the incompressible Navier-Stokes equations [18],

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \nabla \rho \vec{v} &= 0, \\ \frac{\partial}{\partial t} (\rho \vec{v}) + \nabla (\rho \vec{v} \vec{v}) &= -\nabla p + \nabla \bar{\tau}, \end{aligned} \quad (4)$$

where t represents time, \vec{v} is the vector of velocity, p is pressure, and $\bar{\tau}$ refers to the stress tensor.

The shear stress transport (SST) $k-\omega$ turbulence model was selected to model the turbulence terms of the RANS

equations [18]. The SST $k-\omega$ turbulence model is able to model the transport of turbulent shear stress and gives accurate predictions on the onset and amount of flow separation under adverse pressure gradients and has been successfully used in the CFD simulation of complex flows [11, 19–22].

3.2. Computational Domain and Boundary Conditions. The computational domain is a finite space used to simulate the flow around the AUV. In order to minimize the effect of block caused by the AUV, a rectangular domain with a dimension of $8L \times 20D \times 15D$ is selected, resulting in a maximum block ratio of 0.67%, as shown in Figure 4. The length of the domain is $8L$ and the leader AUV is placed in the mid-plane of the domain and $3L$ to the left boundary. The follower AUV is in the same horizontal plane with the leader AUV and is offset in the coordinate frame.

Steady and uniform velocity inlet boundary is set at the left surface of the domain. The magnitude of the inflow velocity is 6 m/s and a moderate turbulence intensity of 5% is chosen. Pressure outlet boundary is chosen for the right surface of the domain, which has the same turbulence intensity with the inlet boundary. Smooth wall conditions are imposed at the four-side surfaces, where the shear effects are neglected to minimize the influences of the walls. Standard wall conditions are applied to the surfaces of both AUVs.

3.3. Mesh Generation. In order to obtain more accurate results with smaller number of grids, the computational domain is meshed with structured hexahedral grids, as shown in Figure 5. Because the flow around the AUVs is more complex and has higher velocity gradients, a higher density of grid is set in this place to capture the development of flow with accuracy. Particularly, prism layer grid elements (Figure 5(c)) are generated from the wall surfaces to refine the mesh quality near the boundary layer flows. The initial height of the prism layer above the wall surfaces is an important indicator for the quality of the mesh. In this study, we use the nondimensional y^+ value to evaluate the quality of the prism layer and a maximum value of 1 is chosen for all meshes. For a smoother

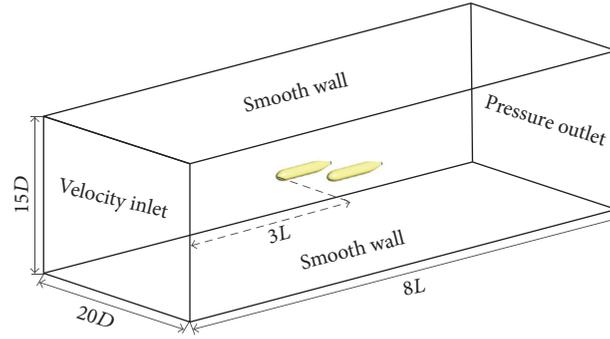


FIGURE 4: Computational domain and boundary conditions.

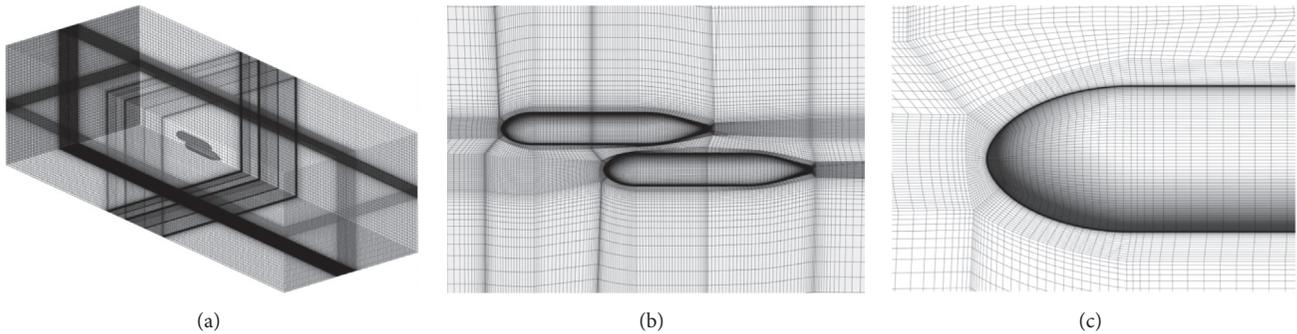


FIGURE 5: Details of the computation mesh: (a) overview; (b) grids around the vehicle; and (c) prism layer grid elements.

TABLE 2: Results of the mesh verification study.

Mesh	Grid number	C_d
Coarse	1670000	0.0786
Mediate	2800000	0.0774
Fine	3760000	0.0771

transition of the adjacent elements, a general mesh growth rate of 1.2 is chosen for all grids.

3.4. Verification and Validation. A mesh verification study is carried out to determine the proper density of mesh. The verification simulations are performed on a single AUV with different meshes. Three meshes, with approximately 3.76 million (fine mesh), 2.80 million (mediate mesh), and 1.67 million (coarse mesh) elements, respectively, are generated for the grid resolution verification. The drag coefficients obtained by the three meshes are listed in Table 2. It is found that the coarse mesh predicts slightly higher C_d and the other two give almost the same results. Therefore, the mediate mesh is chosen for the following simulations. It should be noted that, for the simulations of the AUV fleet, the grid density is kept the same with that of the single AUV and the number of elements is about 5.5 million due to the increased number of AUVs.

A validation study is performed for the validation of the proposed numerical model. Simulations are performed according to an experiment of twin parallel bare prolate

spheroids with transverse separations [16] (a varies and $b = 0$ in current study), as shown in Figure 6. The length of the model was 1.2 m and the maximum diameter was 0.2 m, which is very close to the size of the AUV in this study. The testing inlet velocity was 40 m/s, corresponding to a Reynolds number of 3.2×10^6 according to the length of the model. This Reynolds number was also close to the current simulation. The calculated coefficients of drag and side force of the top spheroid are then compared with those obtained by experiment [16] and a previous CFD study [23] (Figure 7). It is noticed that the current CFD model provides more accurate results than that presented by Molland and Utama [23]. The maximum relative errors of the predicted drag and side force are 3.3% and 9.1%, respectively. This proves that the CFD model in this study can predict the forces of two AUVs with acceptable accuracy.

4. Optimal Design Method

This paper establishes an optimization method based on the combination of CFD method, BPNN and GA. The drag coefficients and the flow structures for 53 different layouts are obtained by the CFD approaches separately. The results are then used to build the agent model between C_d and the design parameters (a and b) using BPNN. Finally, GA is applied to search for the optimal C_d together with the values for the corresponding design parameters based on the agent model.

4.1. BP Neural Network. The artificial neural network is widely accepted as an alternative to providing solutions to

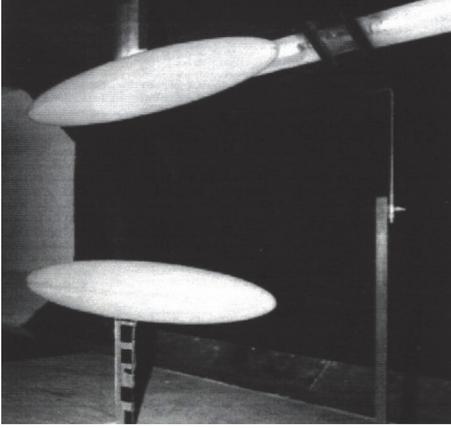


FIGURE 6: Wind tunnel test of the twin spheroids [16].

complex and ambiguous problems [24, 25]. In this study, a back-propagation algorithm that optimizes weighted connections by allowing errors to propagate from the output layer to the input layer is used to train the network. The topology of the BPNN in present study is shown in Figure 8. The structure has two input layers, five hidden layers and one output layer. Each node of the structure represents a neuron, and the nodes between the layers and the layers are connected by weights.

The BPNN is working based on the following principle:

(1) According to CFD simulation results, determine the network input data and output data (X, Y). H is the data matrix of the hidden layer, as can be seen in

$$\begin{aligned} X_p &= (x_{p1}, x_{p2}, \dots, x_{pn}), \\ H_p &= (h_{p1}, h_{p2}, \dots, h_{pl}), \\ Y_p &= (y_{p1}, y_{p2}, \dots, y_{pm}), \end{aligned} \quad (5)$$

where n, m, l , and p are the numbers of inputs, outputs, hidden layers, and the training samples, respectively.

(2) The outputs of the hidden layer are obtained by

$$h_{pi} = f \left(\sum_{i=1}^n \omega_{ij} x_{pi} - a_j \right), \quad j = 1, 2, \dots, l, \quad (6)$$

where ω_{ij} is the connection between the input layer and the hidden layer.

(3) The predicted outputs of the BPNN can be obtained by

$$\hat{y}_{pk} = \sum_{j=1}^l h_{pi} \hat{\omega}_{jk} - b_k, \quad k = 1, 2, \dots, m, \quad (7)$$

where $\hat{\omega}_{jk}$ is the connection between the hidden layer and the output layer.

(4) According to the network forecast output \hat{y}_k and the expected output y_k , calculate the network prediction error e_k .

$$e_k = y_{pk} - \hat{y}_{pk}. \quad (8)$$

(5) The weights are updated according to the network prediction error (e_k).

$$\omega_{ij} = \omega_{ij} + \eta h_{pi} (1 - h_{pi}) x(i) \sum_{k=1}^m \hat{\omega}_{jk} e_k, \quad (9)$$

$$\hat{\omega}_{jk} = \hat{\omega}_{jk} + \eta h_{pj} e_k,$$

where η is the learning efficiency of BPNN.

(6) The thresholds are updated according to the network prediction error (e_k).

$$a_j = a_j + \eta h_{pj} (1 - h_{pj}) \sum_{k=1}^m \hat{\omega}_{jk} e_k, \quad (10)$$

$$b_k = b_k + \eta e_k.$$

(7) Return to step (2) if the optimization is still running.

It is noted that all the iterative processes of the BP neural network have a mean square error (MSE). In addition, the average accuracy of the prediction (R) is defined by Kreith (2000). In other words, R represents the goodness of fit, which is used to measure the correlation between prediction output data and training samples data. The closer it is to the value of 1, the better the training network is.

$$\text{MSE} = \frac{1}{mp} \sum_{p=1}^P \sum_{j=1}^m (\hat{y}_{pj} - y_{pj})^2, \quad (11)$$

$$R = \frac{1}{P} \sum_{i=1}^P R_i = \frac{1}{P} \sum_{i=1}^P \frac{y_{pi}}{\hat{y}_{pi}}.$$

4.2. Genetic Algorithm. In this paper, the GA tool in MATLAB is used for the optimization, which runs in the following principle:

- (1) Initialize the population, calculate the fitness value, and find the best chromosome from the population.
- (2) Iterative optimization:
 - (a) Select: first, the solution to the problem is encoded by using the floating-point encoding. This function selects the chromosomes in each generation population for subsequent crossover and mutation. The method used is the roulette selection method.
 - (b) Crossover: this function is a random selection of two chromosomes, according to determine the crossover probability to determine whether the cross, and the cross position is also random.
 - (c) Mutation: this function performs the mutation operation. The mutation chromosomes and mutated positions are randomly selected. Finally, it will check the feasibility of chromosomes; otherwise, it will be recompiled.
 - (d) Result analysis.

The optimal solution can be found after several generations. The establishment of BPNN approximate model and the numerical optimization of the genetic algorithm are shown in Figure 9.

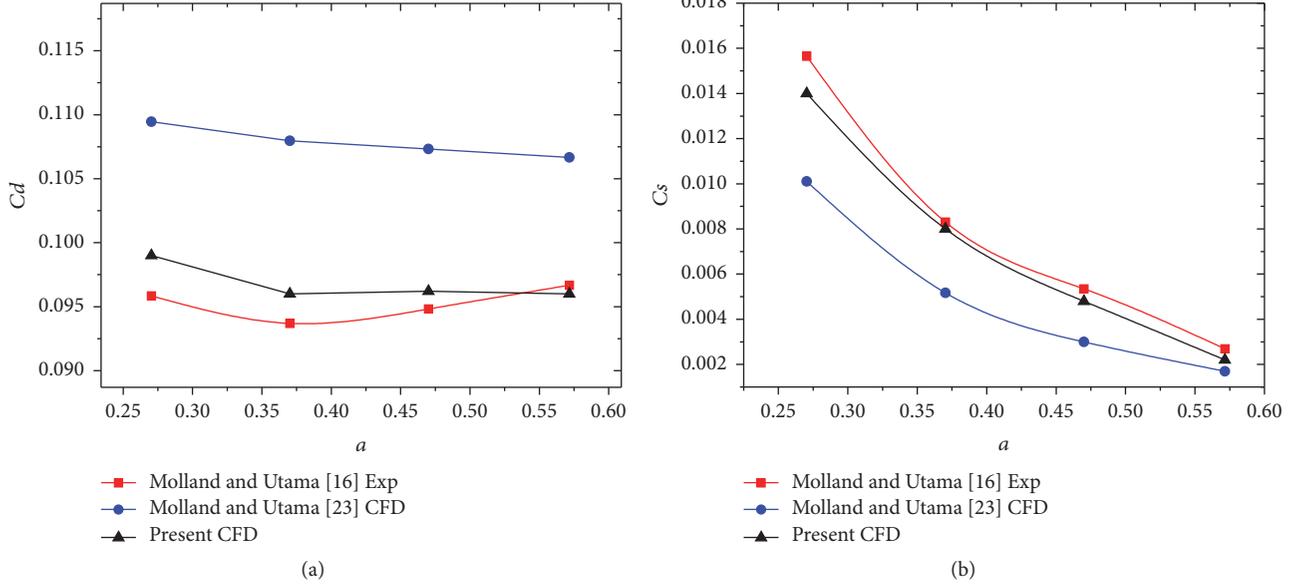


FIGURE 7: Results of the validation study: (a) coefficients of drag and (b) coefficients of side force.

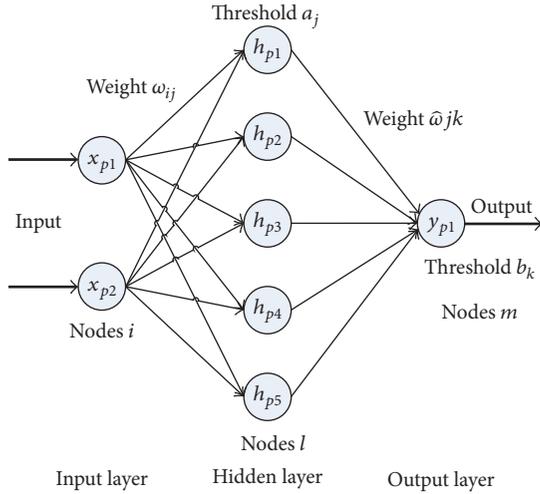


FIGURE 8: The topology of the BPNN.

5. CFD Results

5.1. Force Performance. In order to more intuitively show the influences of the fleet layout on the drags of the two AUVs, the coefficients of drags are expressed in a nondimensional drag ratio by dividing C_{d0} of a single AUV (0.0774 in Table 2):

$$\begin{aligned}
 r_l &= \frac{C_{dl}}{C_{d0}}, \\
 r_f &= \frac{C_{df}}{C_{d0}}, \\
 r_{\text{sum}} &= \frac{C_{dl} + C_{df}}{C_{d0}}.
 \end{aligned} \tag{12}$$

The drag ratios for the 53 different fleet layouts are shown in Figure 10. The configuration of the AUV fleet can be divided into four regions depending on the drag ratios: the Parallel Region, the Pull Region, the Push Region, and the Tandem Region.

5.2. Parallel Region. The Parallel Region locates at $b = 0$. AUVs in this region generate a higher drag than that in an infinite domain. Besides, the drag ratios for both AUVs are almost the same as each other due to the symmetrical geometric positions. For example, the drag ratios of AUVs at $a = 0.20$ and $b = 0.00$ are 5.52% higher than that of a single AUV. It can also be observed that as the latitudinal offset increases, the drag ratios gradually reduce. At $a = 0.60$ and $b = 0.00$, the drags of the AUVs are only approximately 0.6% higher than that of a single AUV. These results are consistent with previous studies [16, 26] which suggested a minimum latitudinal offset of $0.5L$ to preclude the interactions between two AUVs.

The contours of pressure and velocity around the two parallel AUVs are shown in Figure 11. Three different latitudinal offsets, $a = 0.2, 0.4, \text{ and } 0.6$, are presented and compared. It can be seen from the figure that the velocity of the fluid is obviously increased in the range between the two AUVs. According to Bernoulli's Equation, larger fluid velocity corresponds to lower pressure, which can be verified by the pressure distributions in Figure 11(b). For example, in the pressure diagram of $a = 0.2$, the pressure on the nose of the AUV is significantly reduced, which eventually leads to an increase in the drag. As a increases, the interactions between the AUVs get weaker and the drag of the AUV is gradually restored to normal levels.

5.3. Tandem Region. The Tandem Region is where the follower AUV locates just behind the leader AUV ($a = 0$). The

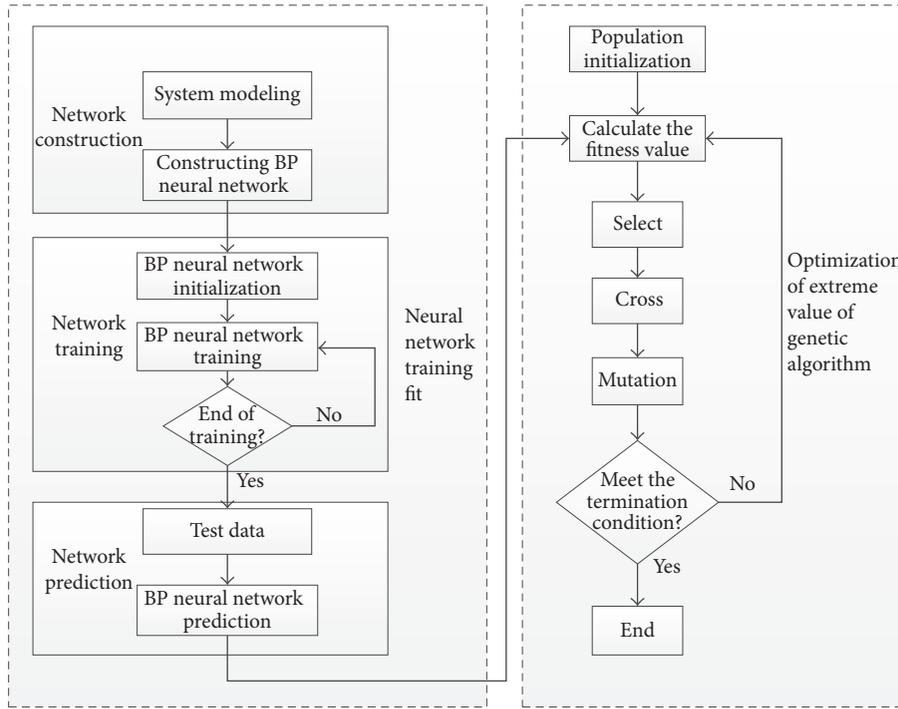


FIGURE 9: Neural network and genetic algorithm flowchart.

	$b = 0.00$	$b = 0.25$	$b = 0.50$	$b = 0.75$	$b = 1.00$	$b = 1.25$	$b = 1.50$	$b = 1.75$	$b = 2.00$	
$a = 0.00$	AUV1/ AUV2					0.7910/ 1.0972	0.9392/ 0.9748	0.9757/ 0.9602	0.9885/ 0.9629	Tandem region
$a = 0.10$						0.8245/ 1.1725	0.9432/ 1.0604	0.9765/ 1.0296	0.9887/ 1.0188	
$a = 0.20$	1.0552/ 1.0552	1.7136/ 0.3598	1.5153/ 0.5364	1.4027/ 0.5841	0.7591/ 1.2507	0.8671/ 1.1107	0.9450/ 1.0398	0.9728/ 1.0150	0.9841/ 1.0050	
$a = 0.30$	1.0247/ 1.0247	1.4041/ 0.6326	1.3595/ 0.6702	1.1989/ 0.7884	0.9257/ 1.0920	0.9135/ 1.0671	0.9557/ 1.0297	0.9761/ 1.0118	0.9852/ 1.0037	
$a = 0.40$	1.0146/ 1.0146	1.2386/ 0.7859	1.2489/ 0.7770	1.1302/ 0.8617	0.9947/ 1.0283	0.9487/ 1.0354	0.9669/ 1.0200	0.9800/ 1.0086	0.9871/ 1.0025	
$a = 0.50$	1.0094/ 1.0095	1.1488/ 0.8699	1.1753/ 0.8506	1.0950/ 0.9000	1.0243/ 1.0032	0.9723/ 1.0146	0.9769/ 1.0118	0.9843/ 1.0057	0.9891/ 1.0016	
$a = 0.60$	1.0062/ 1.0065	1.0963/ 0.9170	1.1263/ 0.8991	1.0724/ 0.9245	1.0724/ 0.9245	0.9873/ 1.0018	0.9850/ 1.0052	0.9883/ 1.0029	0.9911/ 1.0000	
	Parallel region	Pull region				Push region				

FIGURE 10: Drag ratios for different AUV fleet layouts.

drag ratios of the AUVs in this region are greatly reduced, especially for the leader AUV. For example, at $a = 0.00$ and $b = 1.50$, the drag reduction is 6.18% for the leader AUV and is 2.52% for the follower AUV. It should be noted that the follower AUV experiences an 9.72% increase in drag at $a = 0.00$ and $b = 1.25$, but the total drag of the AUV fleet reduces.

The contours of pressure and velocity around the two tandem AUVs are shown in Figure 12. Three different longitudinal offsets, $b = 1.25, 1.50,$ and 1.75 , are presented and

compared. When the follower AUV is located in tandem in the wake of the leader AUV, a certain amount of pressure is recovered on the tail of the leader AUV, which contributes to the reduction of the pressure drag of the leader AUV, while for the follower AUV, the velocity of the upstream flow is reduced by the leader AUV, resulting in a reduction of the nose pressure and a decrease in final resistance.

5.4. Pull Region. The Pull Region is where the drag of the leader AUV is increased and that of the follower AUV is

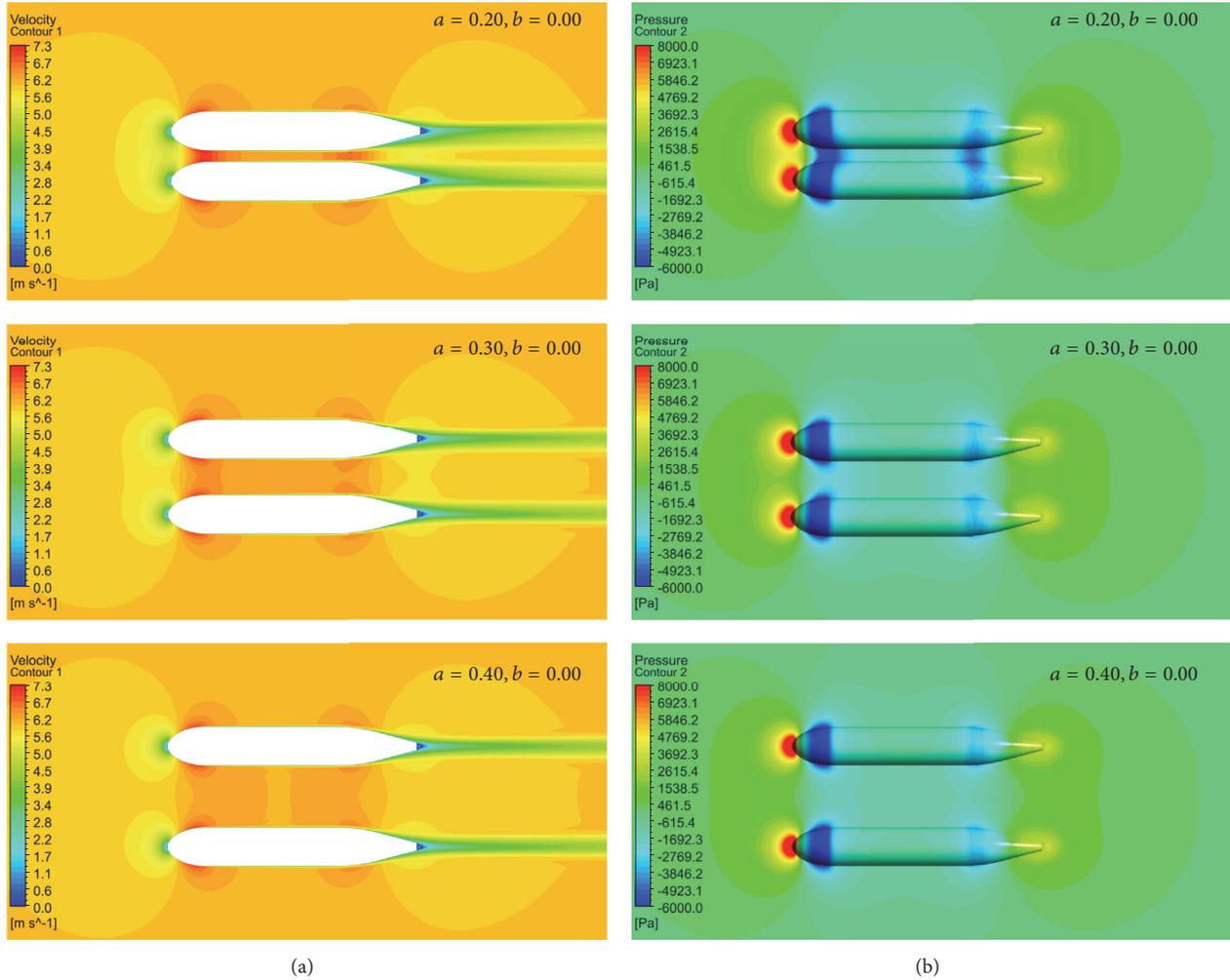


FIGURE 11: Contours of (a) velocity and (b) pressure around two parallel AUVs at different latitudinal offsets.

reduced. The Pull Region mainly locates between $0.20 \leq a \leq 0.60$ and $0.25 \leq b \leq 1.00$. In this region, it is likely that the leader AUV pulls the follower AUV to move, so that the drag of the leader AUV is increased and the drag of the follower AUV changes oppositely.

The contours of pressure and velocity around the two AUVs in the Pull Region are shown in Figure 13. Three different locations with $a = 0.2, 0.3,$ and $0.4, b = 0.25, 0.50,$ and $0.75,$ respectively, are presented and compared. The longitudinal offset is within one AUV length and the nose of the follower AUV locates at the lower pressure region by the side of the leader AUV. Under the effects of the follower AUV, the pressure on the side surface of the leader AUV is reduced, resulting in an augment in the drag. Besides, as the a increases, the interactions between the two AUVs are weaker and the variation in the drags of both AUVs is smaller.

5.5. Push Region. The Push Region is where the drag of the leader AUV is reduced and that of the follower AUV is augmented. The Push Region mainly locates between $0.10 \leq a \leq 0.60$ and $b \geq 1.25$. In this region, it is likely that the

follower AUV pushes the leader AUV to move, so that the drag of the leader AUV is reduced and the drag of the follower AUV changes oppositely.

The contours of pressure and velocity around the two AUVs in the Push Region are shown in Figure 14. Three different locations with $a = 0.2, 0.3,$ and $0.4, b = 1.25, 1.50,$ and $1.75,$ respectively, are presented and compared. The reason for the variation of the drags on the two AUVs can be explained. The obstacle flow around the nose of the follower AUV hinders the separation on the tail of the leader AUV; therefore, a certain amount of pressure is recovered on the tail of the leader AUV, which contributes to a reduction of the pressure drag of the leader AUV. A reaction force is then applied to the follower AUV, leading to an increase in the drag on the follower AUV.

5.6. Data for the BPNN Training Samples. The predicted drag ratios for the AUV fleet obtained by CFD are given in Table 3. It should be noted that 10 samples for $a \leq 0.20$ and $b \leq 1.00$ are added to complete an orthographical sample domain. Since it is nonphysical to arrange the two AUVs in this area, a

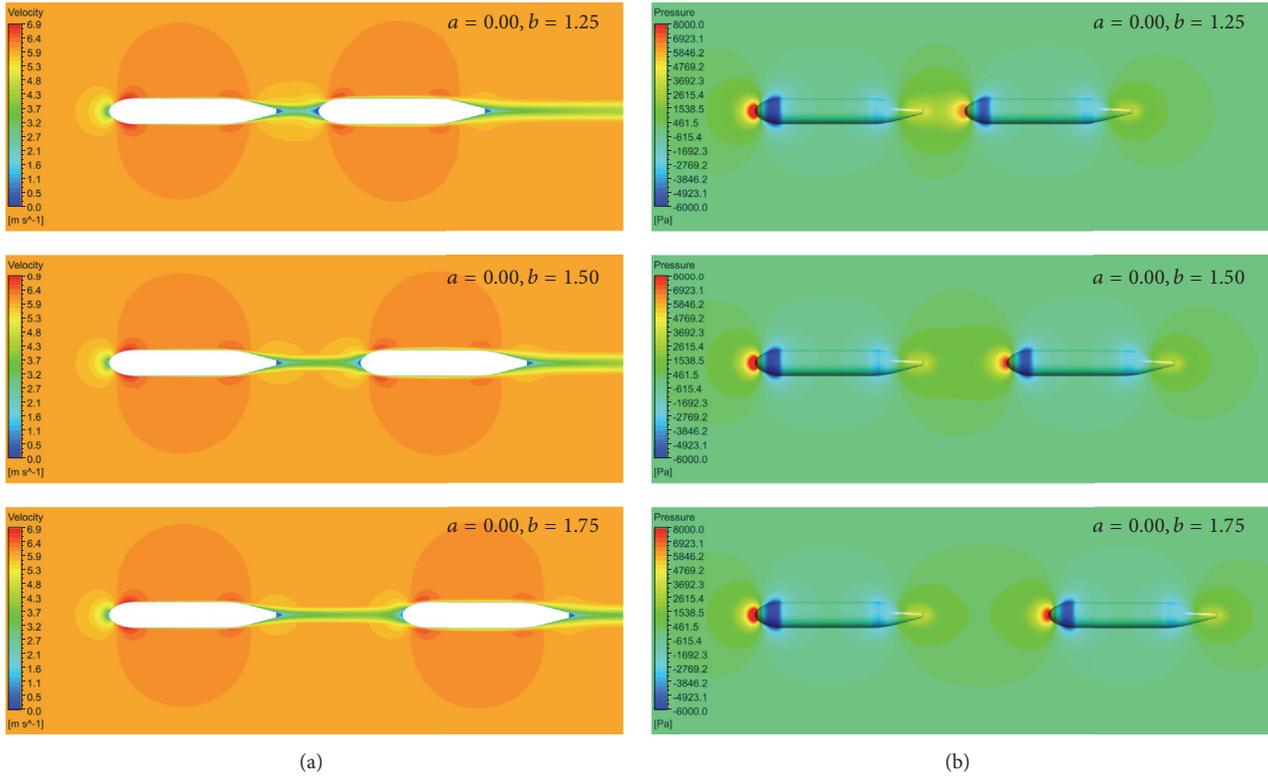


FIGURE 12: Contours of (a) velocity and (b) pressure around two parallel AUVs at different latitudinal offsets.

high drag ratio of 2.0000 is assigned to the AUV which locates in this region. Therefore, there are 63 samples in total for the training of the BPNN.

6. Neural Network and Genetic Algorithm Optimization

6.1. Optimization of the AUV Fleet. When the AUV fleet executes tasks underwater, a minimum drag is expected so that the fleet will work for a longer time and a wider range. Therefore, the layout of the AUV fleet should be optimized to obtain a minimum drag.

6.1.1. Neural Network Training. Using the data in Table 3, training of the BPNN is carried out until the network meets the intended target. A comparison between the predicted data and the expected data is shown in Figure 15. The training network predicts drag ratios with good agreement with the CFD data. Therefore, it can be said that the BPNN successfully model and predict the drag ratio of the AUV fleet.

6.1.2. Comparisons of the Optimal Drag between CFD and BPNN. A CFD simulation of the AUV fleet for the optimal parameters is carried out for the comparisons between the two methods. The optimal results are shown in Table 4. The results of the BPNN and the CFD method are very close, with a relative error of 0.66%, which further validates the accuracy of the BPNN. The optimal drag ratio is obtained at $a = 0.00$

and $b = 1.21$, where the follower AUV is directly behind the Leading AUV, and the optimal drag ratio of the AUV fleet is 1.8825, which means that the total drag of the AUV fleet is reduced by approximately 12%.

To further investigate the flow structures of the optimal fleet layout, Figure 16 compares the pressure contours around the optimal AUV fleet and that of a single AUV. The most obvious difference between these two cases is the pressure distribution on the tail of the leader AUV. Due to the block effect of the follower AUV, the pressure on the tail of the leader AUV is recovered. This variation can be clearly observed in the pressure distributions in Figure 17.

6.2. Optimization of the Follower AUV. Although the optimal fleet layout analyzed in Section 6.2 characterizes with the minimum drag, the drag of the follower AUV is not the optimal value. Among the AUVs in a fleet, it is possible that leader AUV is designed with high propulsive performance and the other follower AUVs obtain lower propulsion costs at the expense of the leader AUV. Therefore, the optimal layout of the fleet for the best performance of the follower AUV should be studied.

6.2.1. Neural Network Training. Using the data in Table 3, training of the BPNN is carried out until the network meets the intended target. A comparison between the predicted data and the expected data for the follower AUV is shown in Figure 18. As can be seen from the figure, the training network predicts drag ratios with good agreement with the CFD data.

TABLE 3: CFD data for the training samples.

Sample	a	b	r_l	r_f	r_{sum}
(1)	0.00	0.00	2.0000	2.0000	4.0000
(2)	0.00	0.25	2.0000	2.0000	4.0000
(3)	0.00	0.50	2.0000	2.0000	4.0000
(4)	0.00	0.75	2.0000	2.0000	4.0000
(5)	0.00	1.00	2.0000	2.0000	4.0000
(6)	0.00	1.25	0.7910	1.0972	1.8882
(7)	0.00	1.50	0.9392	0.9748	1.9141
(8)	0.00	1.75	0.9757	0.9602	1.9359
(9)	0.00	2.00	0.9885	0.9629	1.9514
(10)	0.10	0.00	2.0000	2.0000	4.0000
(11)	0.10	0.25	2.0000	2.0000	4.0000
(12)	0.10	0.50	2.0000	2.0000	4.0000
(13)	0.10	0.75	2.0000	2.0000	4.0000
(14)	0.10	1.00	2.0000	2.0000	4.0000
(15)	0.10	1.25	0.8245	1.1725	1.9970
(16)	0.10	1.50	0.9432	1.0604	2.0036
(17)	0.10	1.75	0.9765	1.0296	2.0061
(18)	0.10	2.00	0.9887	1.0188	2.0075
(19)	0.20	0.00	1.0552	1.0552	2.1104
(20)	0.20	0.25	1.7136	0.3598	2.0733
(21)	0.20	0.50	1.5153	0.5364	2.0517
(22)	0.20	0.75	1.4027	0.5841	1.9868
(23)	0.20	1.00	0.7591	1.2507	2.0098
(24)	0.20	1.25	0.8671	1.1107	1.9778
(25)	0.20	1.50	0.9450	1.0398	1.9848
(26)	0.20	1.75	0.9728	1.0150	1.9879
(27)	0.20	2.00	0.9841	1.0050	1.9891
(28)	0.30	0.00	1.0247	1.0247	2.0494
(29)	0.30	0.25	1.4041	0.6326	2.0367
(30)	0.30	0.50	1.3595	0.6702	2.0297
(31)	0.30	0.75	1.1989	0.7884	1.9873
(32)	0.30	1.00	0.9257	1.0920	2.0177
(33)	0.30	1.25	0.9135	1.0671	1.9806
(34)	0.30	1.50	0.9557	1.0297	1.9853
(35)	0.30	1.75	0.9761	1.0118	1.9879
(36)	0.30	2.00	0.9852	1.0037	1.9889
(37)	0.40	0.00	1.0146	1.0146	2.0292
(38)	0.40	0.25	1.2386	0.7859	2.0245
(39)	0.40	0.50	1.2489	0.7770	2.0259
(40)	0.40	0.75	1.1302	0.8617	1.9920
(41)	0.40	1.00	0.9947	1.0283	2.0229
(42)	0.40	1.25	0.9487	1.0354	1.9841
(43)	0.40	1.50	0.9669	1.0200	1.9869
(44)	0.40	1.75	0.9800	1.0086	1.9886
(45)	0.40	2.00	0.9871	1.0025	1.9896
(46)	0.50	0.00	1.0094	1.0095	2.0189
(47)	0.50	0.25	1.1488	0.8699	2.0187
(48)	0.50	0.50	1.1753	0.8506	2.0259

TABLE 3: Continued.

Sample	a	b	r_l	r_f	r_{sum}
(49)	0.50	0.75	1.0950	0.9000	1.9951
(50)	0.50	1.00	1.0243	1.0032	2.0275
(51)	0.50	1.25	0.9723	1.0146	1.9868
(52)	0.50	1.50	0.9769	1.0118	1.9887
(53)	0.50	1.75	0.9843	1.0057	1.9900
(54)	0.50	2.00	0.9891	1.0016	1.9907
(55)	0.60	0.00	1.0062	1.0065	2.0127
(56)	0.60	0.25	1.0963	0.9170	2.0132
(57)	0.60	0.50	1.1263	0.8991	2.0253
(58)	0.60	0.75	1.0724	0.9245	1.9969
(59)	0.60	1.00	1.0327	0.9907	2.0235
(60)	0.60	1.25	0.9873	1.0018	1.9891
(61)	0.60	1.50	0.9850	1.0052	1.9903
(62)	0.60	1.75	0.9883	1.0029	1.9912
(63)	0.60	2.00	0.9911	1.0000	1.9911

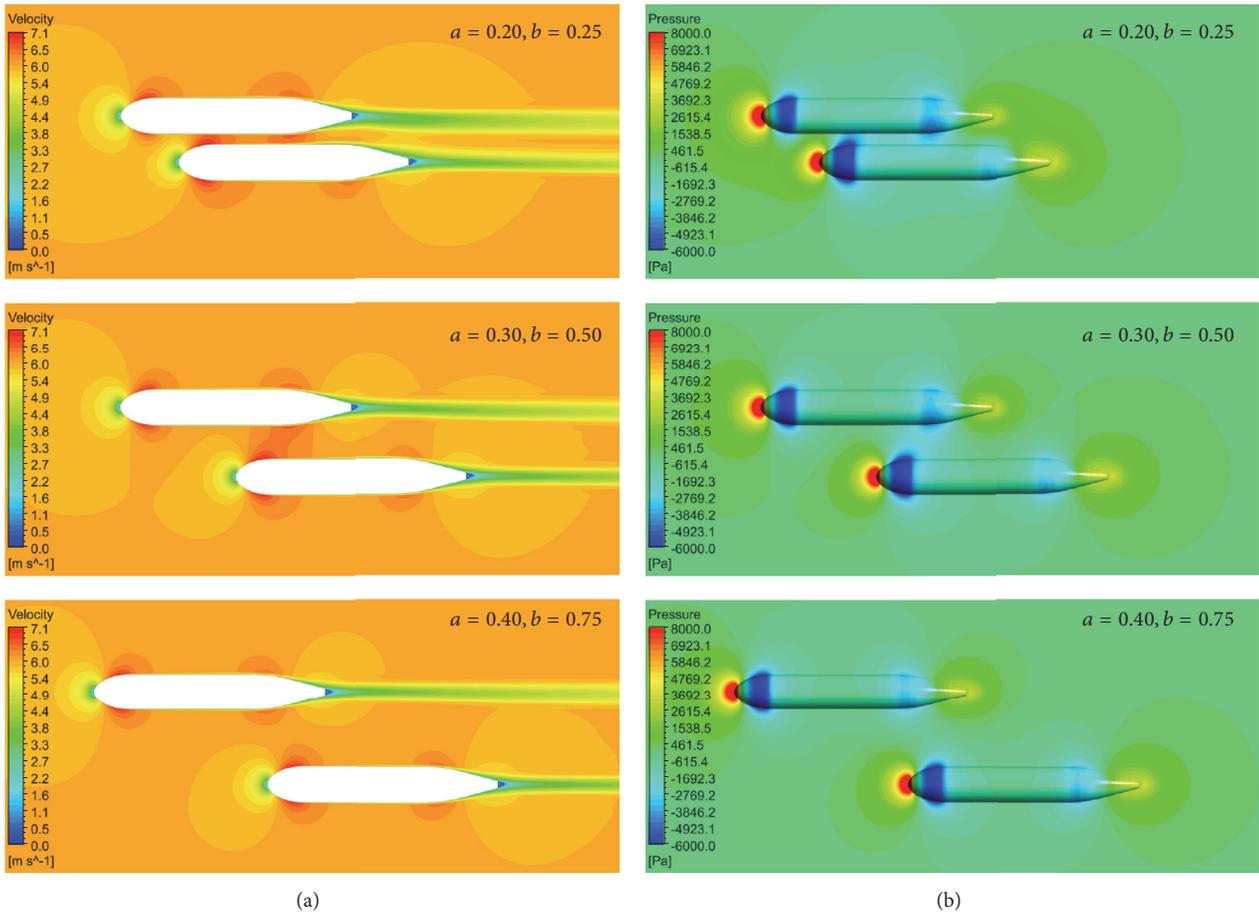


FIGURE 13: Contours of (a) velocity and (b) pressure around two AUVs in the Pull Region.

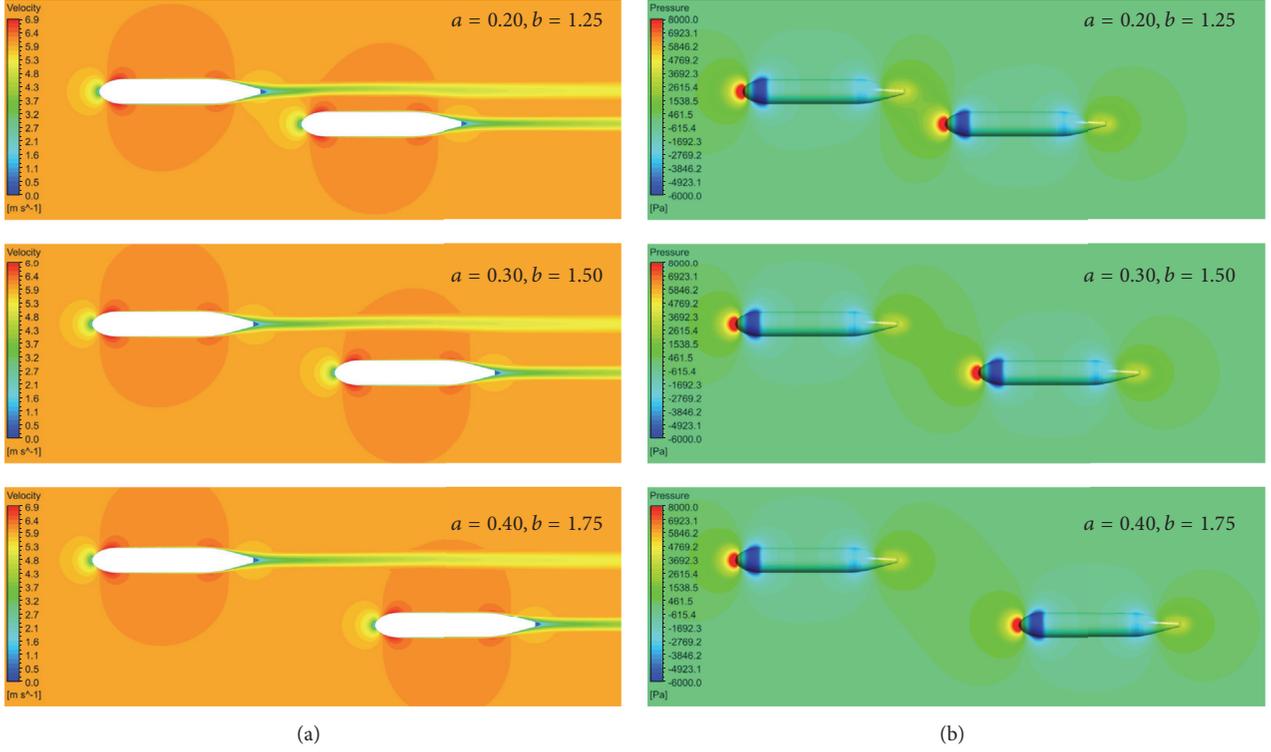


FIGURE 14: Contours of (a) velocity and (b) pressure around two AUVs in the Pull Region.

TABLE 4: Comparisons of the optimal drag ratio of the AUV fleet between CFD and BPNN.

a	b	BPNN	CFD	Relative error (%)
0.00	1.21	1.8949	1.8825	0.66%

TABLE 5: Comparisons of the optimal drag ratio of the follower AUV between CFD and BPNN.

a	b	BPNN	CFD	Relative error (%)
0.22	0.29	0.3481	0.3417	1.87%

6.2.2. Comparisons of the Optimal Drag between CFD and BPNN. Based on the network after the training process, the optimal output drag ratio of the follower AUV is predicted using the BPNN model. Typically, a CFD simulation for the optimal parameters is carried out for the comparisons between the two methods. The optimal results are shown in Table 5. Table 5 suggests that the results of the BPNN and the CFD method are very close, with a relative error of 1.87%, which further validates the accuracy of the BPNN. The optimal drag ratio is obtained at $a = 0.00$ and $b = 1.21$, where the AUV fleet is in the Pull Region. The optimal drag ratio of the follower AUV is 0.3417, which means that the drag of the follower AUV is reduced by approximately 66%.

Figure 19 compares the pressure contours around the optimal AUV fleet and that of a single AUV. The most obvious difference between these two cases is the pressure distribution around the nose of the follower AUV. At this position, the

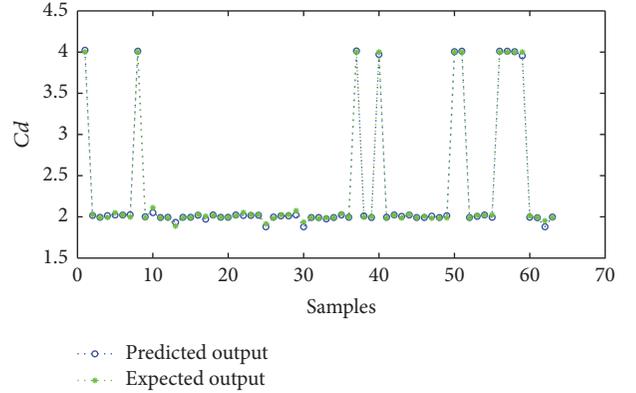


FIGURE 15: Comparisons of the drag ratio of the AUV fleet by BPNN and CFD.

nose of the follower AUV is influenced by the negative pressure on the inside surface of the leader AUV, so that a smaller pressure is obtained. This can also be observed in the pressure distributions in Figure 20.

7. Conclusions

The drag of the AUV fleet significantly influences the operating time and range of underwater vehicles. In this study, an optimization method is proposed to find the optimal layout of the AUV fleet which has the minimum drag. A combined method of CFD simulation, BPNN, and GA method is

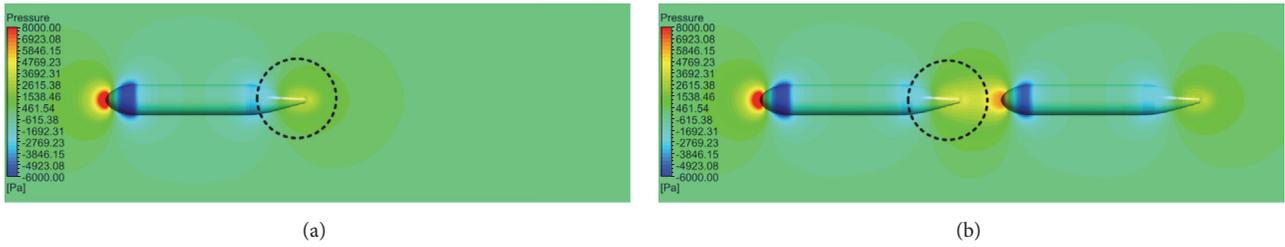


FIGURE 16: Contours of pressure around the AUV: (a) single AUV and (b) optimal AUV fleet.

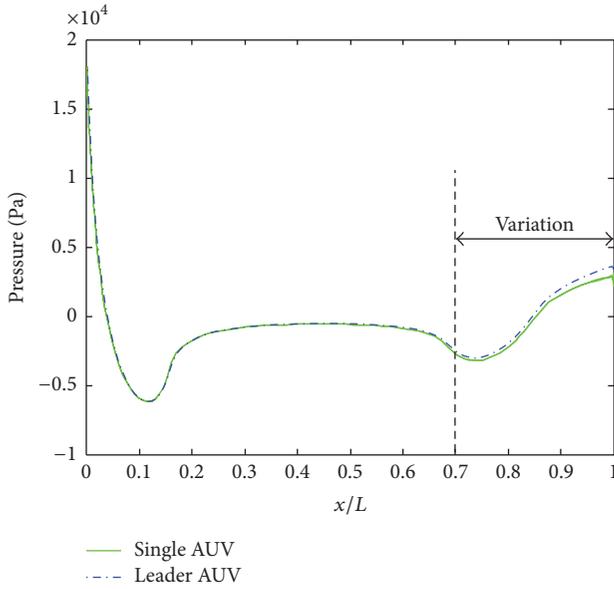


FIGURE 17: Pressure distributions along the longitudinal direction of the AUV: (a) single AUV and (b) optimal AUV fleet.

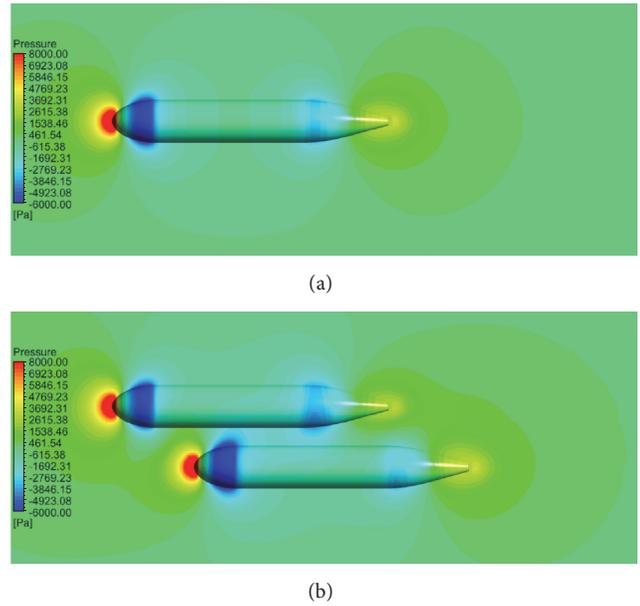


FIGURE 19: Contours of pressure around the AUV: (a) single AUV and (b) optimal AUV fleet.

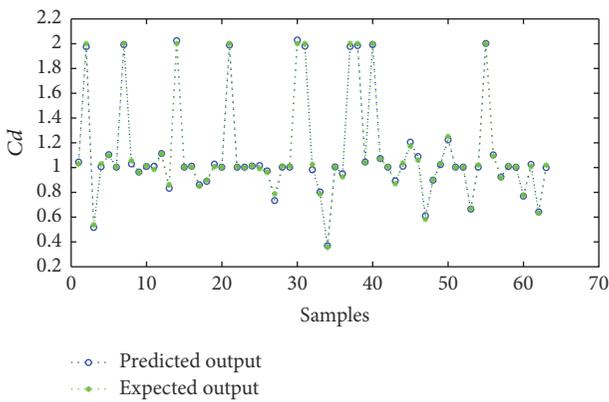


FIGURE 18: Comparisons of the drag ratio of the follower AUV by BPNN and CFD.

utilized for the optimization of the AUV fleet. Important conclusions of this study include the following:

(1) The CFD results show that the drags of the AUVs in the fleet are determined by the relative position of the two AUVs. The layout of the fleet can be categorized into four

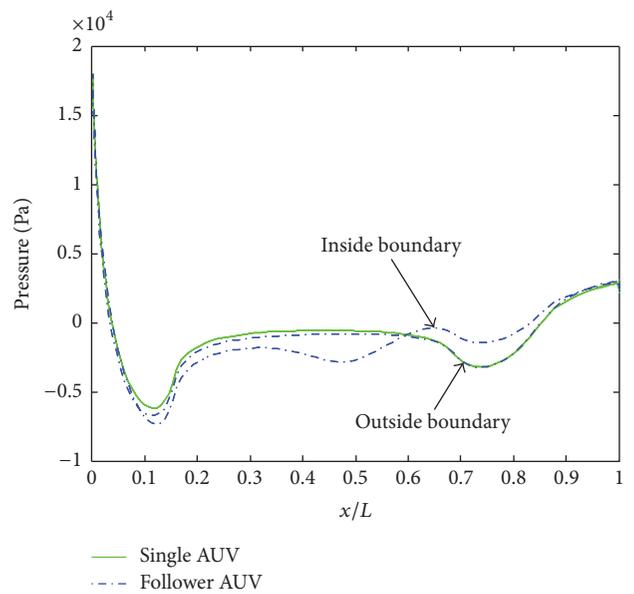


FIGURE 20: Pressure distributions along the longitudinal direction of the AUV: (a) single AUV and (b) optimal AUV fleet.

different regions based on the drag behavior of the two AUVs: the Parallel Region, the Tandem Region, the Pull Region, and the Push Region. The variation of the drags of the AUVs is the result of the interacting flow and pressure change on the two AUVs.

(2) The optimal layout parameters for the minimum fleet drag are $a = 0.00$ and $b = 1.21$, where the fleet is in the Tandem Region and predicts a drag approximately 12% lower than that of a single AUV.

(3) The optimal layout parameters for the minimum drag of the follower AUV are $a = 0.22$ and $b = 0.29$, where the fleet is in the Pull Region and predicts a drag approximately 66% lower than that of a single AUV.

Nomenclature

AUV:	Autonomous underwater vehicles
BPNN:	Back-propagation neural network
CFD:	Computational fluid dynamics
GA:	Genetic algorithm
D :	Maximum diameter of the AUV
L :	Length of the AUV
S :	Maximum cross-sectional area of the (m^2)
U :	Velocity of the AUV
ρ :	Density of seawater
y^+ :	y -plus value
a, b :	Normalized position of the follower AUV
x_f, y_f :	Position of the follower AUV
C_{dl} :	Drag coefficient of the leader AUV
C_{df} :	Drag coefficient of the follower AUV
C_{dsum} :	Drag coefficient of the fleet
r_l :	Drag ratio of the leader AUV
r_f :	Drag ratio of the follower AUV
r_{sum} :	Drag ratio of the fleet
k :	Turbulence kinetic energy
ω :	Specific rate of dissipation.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the National Science Foundation of China (Grants nos. 51179159 and 61572404).

References

- [1] X. Li, H. Zhu, G. Chen, and R. Zhang, "Optimal maintenance strategy for corroded subsea pipelines," *Journal of Loss Prevention in the Process Industries*, vol. 49, pp. 145–154, 2017.
- [2] X. Wang, C. Yang, Z. Ju, H. Ma, and M. Fu, "Robot manipulator self-identification for surrounding obstacle detection," *Multimedia Tools Applications*, pp. 1–26, 2016.
- [3] D. Edwards, T. Bean, D. Odell, and M. Anderson, "A leader-follower algorithm for multiple AUV formations," in *Proceedings of the IEEE/OES Autonomous Underwater Vehicles*, pp. 40–46, Sebasco, Me, USA, June 2004.
- [4] S. Botelho, R. Neves, and B. L. Taddei, "Localization of a fleet of AUVs using visual maps," in *Proceedings of the Europe Oceans 2005*, Brest, France, June 2005.
- [5] R. M. Alexander, "Hitching a lift hydrodynamically—In swimming, flying and cycling," *Journal of Biology*, vol. 3, no. 2, article 7, 2004.
- [6] P. Jagadeesh, K. Murali, and V. G. Idichandy, "Experimental investigation of hydrodynamic force coefficients over AUV hull form," *Ocean Engineering*, vol. 36, no. 1, pp. 113–118, 2009.
- [7] P. Stevenson, M. Furlong, and D. Dormer, "AUV shapes - Combining the practical and hydrodynamic considerations," in *Proceedings of the Europe OCEANS 2007*, pp. 1–6, June 2007.
- [8] A. Alvarez, V. Bertram, and L. Gualdesi, "Hull hydrodynamic optimization of autonomous underwater vehicles operating at snorkeling depth," *Ocean Engineering*, vol. 36, no. 1, pp. 105–112, 2009.
- [9] H. Kim, C. Yang, R. Löhner, and F. Noblesse, "A practical hydrodynamic optimization tool for the design of a monohull ship," in *Proceedings of the 18th International Offshore and Polar Engineering Conference, ISOPE 2008*, pp. 98–107, July 2008.
- [10] T.-H. Joung, K. Sammut, F. He, and S.-K. Lee, "Shape optimization of an autonomous underwater vehicle with a ducted propeller using computational fluid dynamics analysis," *International Journal of Naval Architecture and Ocean Engineering*, vol. 4, no. 1, pp. 44–56, 2012.
- [11] C. Sun, B. Song, P. Wang, and X. Wang, "Shape optimization of blended-wing-body underwater glider by using gliding range as the optimization target," *International Journal of Naval Architecture & Ocean Engineering*, vol. 9, no. 6, pp. 693–707, 2017.
- [12] K. Elsayed and C. Lacor, "The Effect of The Cyclone Separator Cone Height on The Performance Using Artificial Neural Network Model and CFD Simulations," in *Proceedings of the International Symposium on Experimental and Computational Aerothermodynamics of Internal Flows*, Brussels, Belgium, 2011.
- [13] C. Shen, L. Wang, and Q. Li, "Optimization of injection molding process parameters using combination of artificial neural network and genetic algorithm method," *Journal of Materials Processing Technology*, vol. 183, no. 2-3, pp. 412–418, 2007.
- [14] H. Safikhani, A. Abbassi, A. Khalkhali, and M. Kalteh, "Multi-objective optimization of nanofluid flow in flat tubes using CFD, Artificial Neural Networks and genetic algorithms," *Advanced Powder Technology*, vol. 25, no. 5, pp. 1608–1617, 2014.
- [15] H. Avcı, D. Kumlutaş, Ö. Özer, and M. Özşen, "Optimisation of the design parameters of a domestic refrigerator using CFD and artificial neural networks," *International Journal of Refrigeration*, vol. 67, pp. 227–238, 2016.
- [16] A. F. Molland and I. K. A. P. Utama, *Wind Tunnel Investigation of a Pair of Ellipsoids in Close Proximity*, University of Southampton, 1997.
- [17] D. F. Myring, "A theoretical study of body drag in subcritical axisymmetric flow," *Aeronautical Quarterly*, vol. 27, no. 3, pp. 186–194, 1976.
- [18] I. Ansys, "Fluent 15.0 user's guide," 2014.
- [19] W. Tian, Z. Mao, and Y. Li, "Numerical Simulations of a VAWT in the Wake of a Moving Car," *Energies*, vol. 10, no. 4, p. 478, 2017.
- [20] W. Tian, Z. Mao, X. An, B. Zhang, and H. Wen, "Numerical study of energy recovery from the wakes of moving vehicles on highways by using a vertical axis wind turbine," *Energy*, vol. 141, pp. 715–728, 2017.

- [21] Y. Qiu, Z. Liu, X. Chen, and C. Zhan, "RETRACTED ARTICLE: Numerical calculation of maneuvering hydrodynamic forces of drift ship based on SST k-w turbulence model," in *Proceedings of the International Conference on Computer Engineering and Technology*, pp. V5285–V5288, 2010.
- [22] W. Tian, Z. Mao, B. Zhang, and Y. Li, "Shape optimization of a Savonius wind rotor with different convex and concave sides," *Journal of Renewable Energy*, vol. 117, pp. 287–299, 2018.
- [23] A. F. Molland and I. K. A. P. Utama, "Experimental and numerical investigations into the drag characteristics of a pair of ellipsoids in close proximity," *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, vol. 216, no. 2, pp. 107–115, 2002.
- [24] M. Gevrey, I. Dimopoulos, and S. Lek, "Review and comparison of methods to study the contribution of variables in artificial neural network models," *Ecological Modelling*, vol. 160, no. 3, pp. 249–264, 2003.
- [25] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, p. 2004, 2014.
- [26] W.-H. Hucho, "Aerodynamics of road vehicles: from fluid mechanics to vehicle engineering," *Butterworths*, vol. 25, no. 1, pp. 485–537, 1998.

Research Article

Skill Learning for Intelligent Robot by Perception-Action Integration: A View from Hierarchical Temporal Memory

Xinzheng Zhang,¹ Jianfen Zhang,¹ and Junpei Zhong²

¹*School of Electrical and Information Engineering, Jinan University, Zhuhai, China*

²*National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan*

Correspondence should be addressed to Jianfen Zhang; eejfzhang@gmail.com

Received 3 July 2017; Accepted 7 September 2017; Published 15 November 2017

Academic Editor: Yanan Li

Copyright © 2017 Xinzheng Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Skill learning autonomously through interactions with the environment is a crucial ability for intelligent robot. A perception-action integration or sensorimotor cycle, as an important issue in imitation learning, is a natural mechanism without the complex program process. Recently, neurocomputing model and developmental intelligence method are considered as a new trend for implementing the robot skill learning. In this paper, based on research of the human brain neocortex model, we present a skill learning method by perception-action integration strategy from the perspective of hierarchical temporal memory (HTM) theory. The sequential sensor data representing a certain skill from a RGB-D camera are received and then encoded as a sequence of Sparse Distributed Representation (SDR) vectors. The sequential SDR vectors are treated as the inputs of the perception-action HTM. The HTM learns sequences of SDRs and makes predictions of what the next input SDR will be. It stores the transitions of the current perceived sensor data and next predicted actions. We evaluated the performance of this proposed framework for learning the shaking hands skill on a humanoid NAO robot. The experimental results manifest that the skill learning method designed in this paper is promising.

1. Introduction

Skill learning autonomously through interactions with the environment is a crucial ability for intelligent robot, and it improves the flexibility and adaptiveness of robots [1]. Imitation learning is a primary method for implementation the skill learning [2]. A perception-action integration or sensorimotor cycle, as an important issue in imitation learning, refers to information flow from the environment to a sensory and motor structure and back to the environment and sensory inputs. It is the processing of sequential sensor information and their transduction to the successive goal-directed behaviors [3]. Skill learning of intelligent robot actually is a paradigm of learning the links between the perceptual environment inputs and the feedback action system. Perception-action integration is a natural mechanism for skill learning without the complex program process. Several research results from cognitive science provide convincing evidence for this statement. Mental simulation [4] stemming from neuropsychology is treated as part of the reactive and

perceptual components because it implements interaction between the action and sensed state of the environment. Perceptual-motor theory [5] states that a neurocomputational framework is used to connect with up-to-date perceptual data on the possible functional role of the motor system. Wolpert et al. [6] reviewed on the computational mechanisms of sensorimotor learning. Also, the computational models of perception-action integration are the dominant technique of skill learning in robotics research field.

1.1. Related Work. Traditional computational model of artificial intelligence methods such as Bayesian modeling [7] and reinforcement learning [8] were first concerned for learning skill. Recently, neurocomputing model and developmental intelligence method are considered as a new trend for implementing the robot skill learning. Do et al. [9] proposed a bootstrapping method for learning the wipe skill. This method bootstrapped from the sensorimotor experience and learned the association between object properties and action parameters. PerAc neural network [10] is applied to learn

the dynamics of coupling of perception of one partner to the action of the other, and the learned association of perception and action is used for recognizing the postures. Neural motor activation [11] that mimics the neuron activation process assigns a weight to each motor component to reveal its degree of activation. The weights are updated by the perception procedure. Neural activation is induced by perception processes with system feedback; therefore, it realizes integration between perception and action. A neural network architecture combining a recurrent neural network with parametric biases (RNNPB) and a horizontal product model was used to predict future percepts and behaviors of a sensorimotor system according to the connections between the development of the ventral/dorsal visual streams and the emergence of conceptualization in the visual streams [12].

Furthermore, some complex cognitive architectures used to simulate the brain pathways of perception-action cycles have been studied. Cutsuridis and Taylor examined several neurocomputational mechanisms of visuomotor brain processes and coordinately integrated them to establish a neural framework for the visual grasping tasks [13]. A cognitive model [14] based on Skinner operant conditioning principle is designed for a robot to master the balancing skill. This model consists of cerebellum, basal ganglia, and cerebral cortex. Each component imitates the basic functions of corresponding parts in human brain. In particular, cerebellum maps sensorimotor states to actions with supervised learning and basal ganglia provide the proper action based on the operant conditioning theory.

1.2. Why Hierarchical Temporal Memory (HTM). As futurist Kurzweil described in his book [15], the neocortex contains a hierarchy of pattern recognition circuits and they are responsible for most aspects of human thought. He also explains that if there exists a design of the digital neocortex, it could be used to create the same capabilities as the human brains. Hierarchical temporal memory (HTM) theory [16], first proposed by Hawkins [17], is an implementation version of Kurzweil’s view of digital neocortex. It attempts to model the brain at a functional level rather than at a neuron or molecular level. HTM is a bioinspired model that captures the predominant characteristics of the neocortex. It mimics the neocortex’s abilities of learning, inference, and prediction from sequential input patterns that are represented in sparse distributed forms, and therefore, it can describe a complex model of the world. Additionally, HTM uses the Sparse Distributed Representations (SDRs) to represent the complex input data and lend the HTM so much flexibility, which is similar to the idea that the brain is a recursive probabilistic fractal whose line of code is represented within the 30–100 million bytes of compressed code in the genome [15].

The cells (neurons) in HTM participate in the sensorimotor integration and learning process, which is supported by biological evidence [18]. In addition, the Cortical Learning Algorithm (CLA) of HTM consists of spatial pooling and temporal memory processes. These are the important components for perception-action integration, which is proved by Lalanne and Lorenceau [19]. Also spatial and temporal capabilities facilitate the acquisition of sensory-motor mappings

with less amount of training data and facilitates the robust behavior [20]. In that research work, they stated that the brain utilizes spatial and temporal coincidence from spatial information when spatial features gathered through different modalities are interconnected.

The core of Kurzweil’s book is the pattern recognition theory of mind. Its main idea is that the hierarchical structure is treated as pattern recognizer and is not just for sensing the world, but for nearly all aspects of thought. It is natural that HTM was first successfully applied for pattern recognition system [21–23].

The reasons and relationships between HTM and neuroscience stated above indicate that HTM can be considered as a promising approach for the implementation of perception-action integration. Therefore, in this study, we applied HTM to design a perception-action integration framework for skill learning. This framework receives the sequential sensor data representing a certain skill from a RGB-D camera. These sensory data are then encoded as a sequence of Sparse Distributed Representation (SDR) vectors. The sequential SDR vectors are treated as the inputs of the perception-action HTM. The HTM learns sequences of SDRs and makes predictions of what the next input SDR will be. It stores the transitions of the current perceived sensor data and next predicted actions. We evaluated the performance of this proposed framework for learning the shaking hands skill on a humanoid NAO robot.

2. HTM Preliminary

The purpose of this study is to implement the skill learning by using HTM based perception-action integration framework, and the process of design follows the general HTM workflow illustrated in Figure 1. The network learns from the time-varying inputs. In this application, the inputs are the captured skeleton joints and depth data. These inputs are encoded by an encoder [16] as a sparse binary string or matrix, which is the necessary input form for an HTM system. In our case, sequential joints data for shaking hands accompanied with depth data are recorded and encoded as a 1,024-bit binary string by an encoder. The format of this string is as

$$\begin{array}{cccccccc} \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{\times \dots \times}^{32 \text{ bits}} & \dots & \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{\times \dots \times}^{32 \text{ bits}} & \overbrace{0 \dots 0} & & \\ \text{joint 1} & \text{joint 2} & & \text{joint } n & \text{depth} & \text{reserved} & & \end{array} \quad (1)$$

The HTM system learns continuously with the input data. The learning algorithms CLA are designed to work with sensor and motor data that is constantly changing. With each change in the inputs, the memory of the HTM system is updated. The HTM uses the CLA dynamic process to learn the spatial and temporal variability commonly occurring in sequential input data and then to make predictions. The typical CLA is composed of two subprocesses: spatial pooling (SP) and temporal memory (TM) algorithms.

Inputs coming from the senses or other parts of the HTM are messy and irregular. The most fundamental function of the SP algorithm is to convert a region’s input into an SDR via overlapping computation, inhibition, and update processes while retaining semantically encoded information. Each SDR

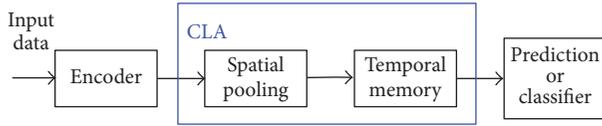


FIGURE 1: Workflow of an HTM application.

has semantic attributes of what is being represented. By determining the overlap between any two SDRs we can immediately see how two representations are semantically similar and how they are semantically different. Because of this semantic overlap property of SDRs, SP associatively connects the input to the HTM cells in a way that they will be able to learn once patterns in the space start to change. TM algorithm is a memory of transitions in a data stream. It learns sequences of SDRs formed by the SP algorithm and makes predictions of what the next input SDR will be. TM is used in both sensory inference and motor generation. It forms a representation of the sparse input that captures the temporal context of previous inputs and then forms a prediction based on the current input in the context of previous inputs. HTM theory postulates that every excitatory neuron in the neocortex is learning transitions of patterns and that the majority of synapses on every neuron are dedicated to learning these transitions.

As a memory system, HTM is essentially a type of neural network. It models the cells, connects and arranges cells in columns, organizes columns in a 2D array to constitute the HTM region, and finally establishes a hierarchical neural network, which is shown in Figure 2. The detailed explanation, properties, and learning pseudocode of HTM can be found in technique reports [16]. In this section, we only describe the crucial contents related to our application.

2.1. Cells. HTM cells extract the most important capabilities of biological neurons, and as shown in Figure 3, they have more complex structures than conventional artificial neurons. A typical HTM cell has three output states: the active state activated from feed-forward input, the predictive state activated from lateral input, and the inactive state. Each HTM cell in one column shares a single proximal dendrite segment (closest to the cell body) and has a list of distal dendrite segments (farther from the cell body). The proximal dendrite segment receives all feed-forward inputs, including the environmental sensory data and outputs of the lower-level region, via active synapses marked by green dots. These active synapses have a linear additive effect at the cell body. Distal dendrite segments receive the lateral inputs from nearby cells through active synapses marked by blue dots. Figure 3 shows that each distal dendrite segment is a threshold detector. The segment will be activated if the number of active synapses on a segment is above a threshold Th_{seg} . An OR operation is executed on all active distal dendrite segments to make the associated cell become the predictive state. Synapses of the HTM cells have binary weights and are formed by a set of potential synapses. The potential synapses are axons that are sufficiently close to a dendrite segment and may become

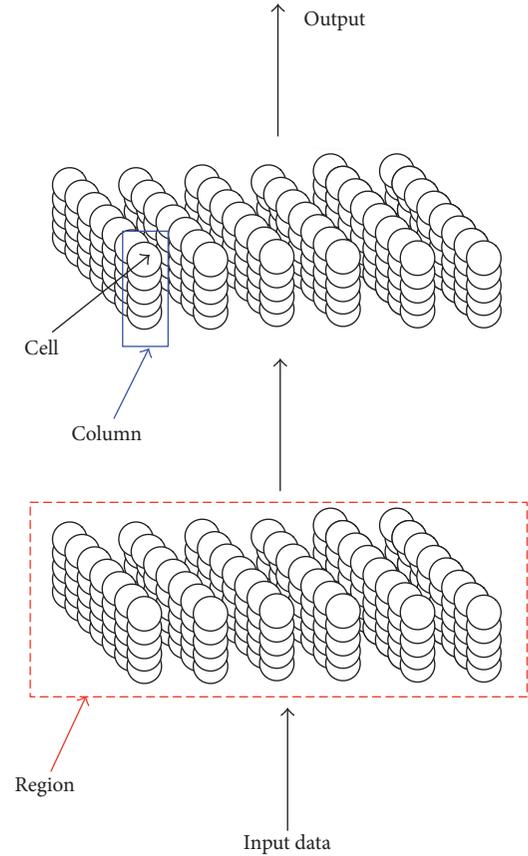


FIGURE 2: The structure of a typical HTM network.

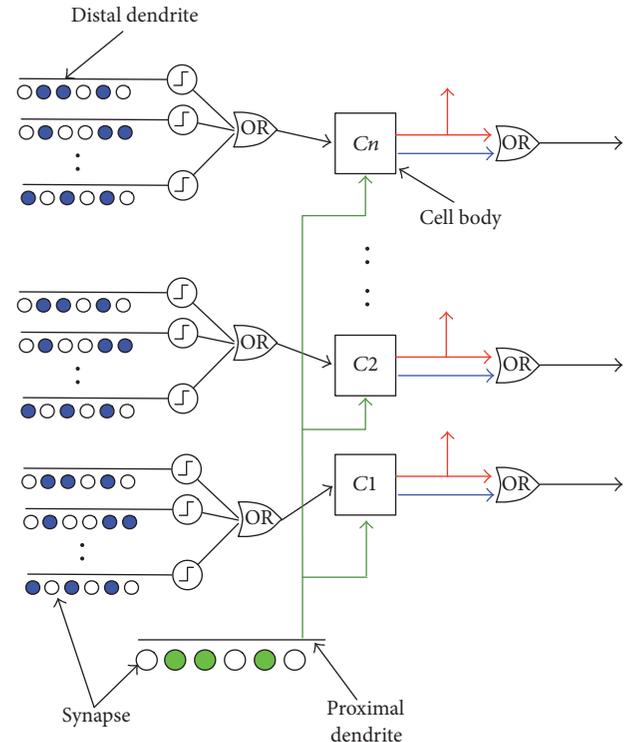


FIGURE 3: The structure of an HTM cell.

synapses. For the proximal dendrite, a potential synapse consists of a subset of all inputs to a region, and for the distal dendrite, the potential synapses are predominantly from the nearby cells in a region. Each potential synapse is assigned a scalar value ranging from 0 to 1. This scalar value is named as permanence, which represents a closeness or connection degree between an axon and dendrite segment. A larger permanence yields a stronger connection. If the permanence is above a threshold Th_{per} , the potential synapse becomes a valid synapse, and the weight of this valid synapse is set as 1. The cell body receives the inputs of synapses from proximal and distal segments and provides two outputs along the axon: one is in an active state (red lines), which is horizontally sent to other adjacent cells, and the other is the OR results of the active and predictive states (blue lines) and sent to the cells of the next region. Because the perception and action are integrated in the HTM network, distal dendritic input can also be the external input. That is, lateral connections between cells will typically be turned off in sensorimotor inference.

2.2. Spatial Pooling (SP). The essential function of spatial pooling is to form an SDR of the inputs. When an input appears on a region, each bit in the input signal will be assigned only to a fixed number of columns. Each column has an associated proximal dendritic segment (shared by all cells of a column, cf. Figure 3), serving as the connection to the input space. Each proximal dendrite segment has a set of potential synapses representing a subset of the input bits. Each potential synapse has a permanence value. These values are randomly initialized around the permanence threshold. Based on their permanence values, when the permanences are greater than the threshold value $Th_{syn,per}$ some of the potential synapses will already be connected.

For any given input, determine how many connected synapses on each column are connected to active input bits (bit 1). The connected synapses become active synapses. The number of active synapses is multiplied by a boost factor bf , which is dynamically determined by how often a column is active relative to its neighbors. The columns with the highest activation after boosting disable a fixed percentage of the columns within an inhibition radius. The result of the inhibition is to form a sparse set of active columns that are treated as the inputs of the TM subprocess in the same region. A Hebbian-like learning procedure is implemented for each of the active columns. Permanence values of synapses aligned with active input bits are increased, and those aligned with inactive input bits are decreased. The changes of permanence values make some synapses become valid or invalid accordingly. Simultaneously, the boost factor and inhibition radius are both updated according to

$$\begin{aligned} bf_c &= f_{bf}(ADC_{avg}^c, ADC_{min}^c) \\ r_{inh} &= \frac{CS_{avg} \times PI_{col} - 1}{2}, \end{aligned} \quad (2)$$

where ADC_{avg} (active duty cycle) is a sliding average that represents how often column c has been active after inhibition, for example, over the last 500 iterations. ADC_{min}

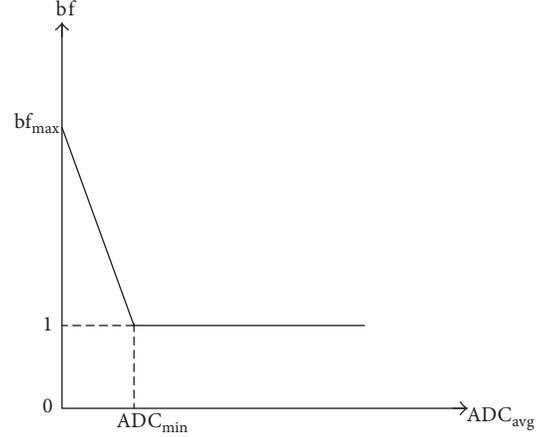


FIGURE 4: Function for updating the boost factor.

represents the minimum desired firing rate for column c . f_{bf} is the update function, which linearly interpolates the boost factor between the points $(0, bf_{max})$ and $(DC_{min}, 1)$, as shown in Figure 4. In general, the boost factors for all columns are updated simultaneously. For the inhibition radius updating, the number of inputs to which a column is connected (denoted by CS_{avg}) should first be determined, and then, this number is multiplied by the total number of columns that exist for each input (denoted by PI_{col}). For multiple dimensions, the aforementioned calculations are averaged over all dimensions of inputs and columns.

2.3. Temporal Memory (TM). TM is more complex than SP because it combines the learning and prediction procedures. It learns SDRs formed by the SP algorithm and makes predictions. TM consists of three phases.

2.3.1. Phase 1: Forming a Representation of the Input in the Context of Previous Inputs (Determining the Active State of Cells). After spatial pooling, the TM algorithm converts the columnar representation of the input into a new representation that includes state, or context, from the past. The new representation is formed by activating a subset of the cells within each column, typically only one cell per column.

For each active column obtained in SP, the cells that are fired to a predictive state from a previous time are activated (referring to (3)). Simultaneously, the distal dendrite segment on each of these cells is marked as active when the number of synapses is over a threshold Th_{act} . The learning cells are chosen by (6). Additionally, if a segment is activated from the learning cells during the previous time, the cell to which this segment connects is set as the learning cell (see (4)). If no cell is in a predictive state, all of the cells in the column are activated, which is defined in (5). For this case, the segment that has the largest number of active synapses is found in column c cell i at time $t - 1$, and then, the related cell to which this segment connects is chosen as the learning cell. If no cell has such a segment, we select the cell that has the fewest number of segments as the learning cell (see (6)). In

Phase 1, the resulting set of active cells consists of the current input in the context of prior inputs.

For the perception-action integration case, there is an optional “Learn-On-One-Cell (LOOC) (Available at <https://github.com/numenta/htmresearch/wiki/Sensorimotor-Inference-Algorithm>)” hysteresis mode. This mode is switched in the following situation. When a column is not predicted but activated by the sensory input, cells that were previously selected as the learning cell would still act as the learning cell at the current time. If no such cell exists, the learning cell is also determined by (6). If the LOOC mode is triggered, a copy of the motor signal is added to the input of the distal dendrites.

$$\begin{aligned} na_i^c(t) &= 1, \\ c \in C_{act}(t) \\ \\ nl_i^c(t) &= 0, \\ c \in C_{act}(t) \end{aligned} \quad (3)$$

$$\text{if } np_i^c(t-1) = 1, \quad sga_i^c(t-1) = 1$$

$$\begin{aligned} na_i^c(t) &= 1, \\ c \in C_{act}(t) \\ \\ nl_i^c(t) &= 1, \\ c \in C_{act}(t) \end{aligned} \quad (4)$$

$$\text{if } np_i^c(t-1) = 1, \quad sga_i^c(t-1) = 1, \quad sgl_i^c(t-1) = 1$$

$$na_i^c(t) = 1, \quad i = 1, \dots, n_c \quad (5)$$

$$nl_i^c(t) = 1, \quad c \in C_{act}(t) \quad (6)$$

Equation (6) is subject to the condition that “(cell i has the segment with the largest number of active synapses at time $t-1$) || (cell i with the fewest number of segments if at time $t-1$).” $na_i^c(t)$ represents the active state of cell i in column c at time t given the current feed-forward input and previous temporal context; $nl_i^c(t)$ and $np_i^c(t-1)$ are the learning and predictive state of cell i in column c at time t and $t-1$, respectively; and $sga_i^c(t-1)$ represents the active segment on cell i in column c at time $t-1$. Similarly, $sgl_i^c(t-1)$ is the segment activated by the learning cell at time $t-1$. If multiple segments are active, sequence segments are given preference. n_c is the number of cells in column c . $C_{act}(t)$ is the set of the active column index at time t .

2.3.2. Phase 2: Forming a Prediction Based on the Input in the Context of Prior Inputs. Following *Phase 1*, according to (7), the cells with active segments are admitted to the predictive state unless they are already active due to feed-forward input. $np_i^c(t)$ represents the predictive state of cell i in column c at time t . All of the predictive cells form the prediction of the region.

$$np_i^c(t) = 1, \quad \text{if } sga_i^c(t) = 1. \quad (7)$$

On column c cell i , the current active segment is added to the update list $SU_i^c(t)$, which will be used in *Phase 3*. To extend

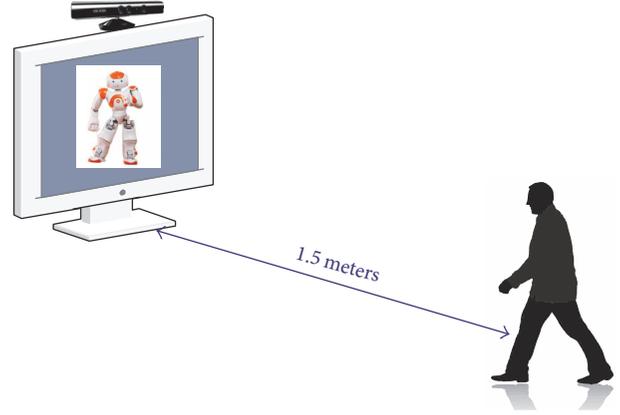


FIGURE 5: Settings for performance examination.

the prediction back in time, another distal dendrite segment that has the largest number of active synapses at the previous time is also considered to add to the update list.

2.3.3. Phase 3: Updating Synapses. Similar to the synapse updates of the proximal dendrite in the SP algorithm, whenever a distal dendrite segment becomes active, the permanence values of its associated potential synapses are modified by the Hebbian rule only if the cell correctly predicted the feed-forward input. Thus, the synapse permanence values for the segments in update list $SU_i^c(t)$ will be reinforced positively or negatively.

Finally, a vector representing the OR results of the active and predictive states of all cells in a region becomes the input to the next region in the hierarchy. Rather than storing a set of predicted cells, TM algorithm stores a set of active distal dendritic segments, that is, the segments related to predicted skeleton positions for the shaking hands. With the prediction, the HTM network can estimate approximately when the inputs will likely arrive next and invoke and separate the motor information.

3. Results

3.1. Experimental Setup. We applied the HTM based perception-action integration for learning the shaking hands skill on a NAO robot. Since there is no practical NAO robot in our lab, the Webots NAO simulator combined with the Kinect V1 RGB-D cameras is considered as the experimental configuration for examining the skill learning performance. As is shown in Figure 5, the RGB-D camera installed on the top of LCD is for simulating the camera and sonar sensors of the real NAO robot. The RGB-D camera captured sequential human’s motion skeletons and depth data between the NAO simulator and a human. It should be noted in Figure 5 that the distance between camera and the object has to be configured within the effective detection range of RGB-D camera, that is, 2 meters for Kinect V1. Here, we set 1.5 meters.

To learn the skill for NAO robot, we need the perception and action data. Therefore, we collected the training data

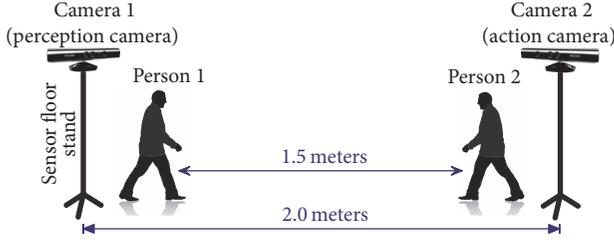


FIGURE 6: Settings for training data collection.

from two persons. These training data are used to swarm the best HTM model parameters. The setting for training data collection is shown in Figure 6. One person stood 1.5 meters far from another one, and two RGB-D cameras were installed on top of their heads. The distance between two sensor floor stands is 2 meters. Camera 1, named as perception camera, is used to acquire the skeleton data of person 2 and the depth from person 2. The purpose of camera 2 (i.e., action camera) is the same as camera 1 except that the data are from person 1. Note that the depth data from the separate cameras have to be converted to the distance between two persons. Combining the skeleton data from two cameras and the converted distance, we build up the training dataset. Two groups of training data were recorded, and each group consists of ten sets of shaking hands skeleton data and depth data. Group one is for the case that one person (person 2) walked towards and stopped 0.5 meters far from another one (person 1) and then shook hands. Group two is for the case that two persons walked towards and stopped 0.5 meters far from each other and then began to shake hands. Camera 1 captured skeleton data of person 2 which will be treated as the perception data for NAO simulator; camera 2 recorded skeleton data of person 1 which is to be taken as the action data for NAO simulator. Because these perception-action skeleton motion data are from different cameras, it is necessary to consider the synchronization issue of data acquisition time. In this paper, we applied asynchronous mechanism to address this problem. The perception camera acquisition thread first started

and then triggered the action camera acquisition thread. These two threads are alternate. Furthermore, when two data acquisition threads started, persons stood statically 5 seconds to maintain the skeleton data stable before recording. This asynchronous manner imitates the perception-action cycle. When the HTM network is trained, we used the experimental setup in Figure 5 to examine the performance of skill learning in online form. The RGB-D camera captured the first frame of person's skeleton and measured the distance between the simulator and the person. These perception data were sent to HTM network, and HTM network provided the predicted skeleton actions. The predicted skeleton actions are converted from the skeleton coordinates system of RGB-D camera to the joint coordinate framework of NAO simulator and then executed on the joints. This is a perception (sensor data acquisition) action (prediction) cycle in online evaluation. This cycle is performed frame by frame until the hand shaking is completed.

The data structure of the recording file is as (8). The data are ordered following the time stamp. "ID" depicts the RGB-D camera ID which captured the skeleton data; depth data are always acquired only by camera 1 (ID = 1) and for the depth bits of camera 2 we copied the depth data of camera 1 directly. It can be found that the perception and action data are recorded alternately. Each joint data consist of 3D Cartesian coordinates and 20 joints include 60 coordinate values (referring to https://msdn.microsoft.com/en-us/library/nui-skeleton.nui_skeleton_position_index.aspx). Practically, for NAO simulator, there are only 12 joints that can be controlled and each joint comprises several Euler angle values (referring to Table 1). It is necessary to address this issue before the swarming process. We selected the corresponding skeleton joints and converted their 3D Cartesian coordinates to Euler angles and then reorganized the converted data following the format of (9) for the optimal parameters swarming. Therefore, 12 joints of NAO simulator cover 20 Euler angle values. Since the size of image is 640×480 , to make the computation efficient, the depth information within a region of interest (ROI) was extracted. The ROIs were selected as a 64×48 rectangle around the image center. The sampling time is set as 100 milliseconds.

$$\begin{array}{ccccccc}
 \text{TimeStamp} & \text{ID} & \text{Joint}_1 & \text{Joint}_2 & \cdots & \text{Joint}_{20} & \text{depth} \\
 15 : 26 : 46 : 151 & 1 & (x, y, z)_1 & (x, y, z)_2 & \cdots & (x, y, z)_{20} & d_1 \\
 15 : 26 : 46 : 198 & 2 & (x, y, z)_1 & (x, y, z)_2 & \cdots & (x, y, z)_{20} & d_1 \\
 \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots
 \end{array} \tag{8}$$

$$\begin{array}{ccccccc}
 \text{TimeStamp} & \text{ID} & \text{Joint}_1 & \text{Joint}_2 & \cdots & \text{Joint}_{12} & \text{depth} \\
 15 : 26 : 46 : 151 & 1 & \text{Euler}_1 & \text{Euler}_2 & \cdots & \text{Euler}_{12} & d_1 \\
 15 : 26 : 46 : 198 & 2 & \text{Euler}_1 & \text{Euler}_2 & \cdots & \text{Euler}_{12} & d_1 \\
 \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots
 \end{array} \tag{9}$$

The HTM was designed based on the open source NuPIC (available at <https://github.com/numenta/nupic>), and

its settings were identical for both of the two cases above. The HTM model is one-region network. The size of the columns

TABLE 1: The Euler angles of NAO simulator joints.

Joint	Euler angle
Left shoulder	Pitch & roll
Left elbow	Yaw & roll
Left wrist	Yaw
Right shoulder	Pitch & roll
Right elbow	Yaw & roll
Right wrist	Yaw
Left hip	Pitch & roll
Left knee	Pitch
Left ankle	Pitch & roll
Right hip	Pitch & roll
Right knee	Pitch
Right ankle	Pitch & roll

in this region is set to 2,048 (arranged as 64×32 in a 2D plane), and the number of cells in each column was set to 32. This configuration maintains the diversity of SDRs and a low probability of a false match between any two SDRs. As is shown in (1), the converted skeleton data are encoded as a binary string by a scalar encoder and each joint data occupied 32 bit. The depth data are also encoded as a 32-bit binary string by a category encoder. Here we defined two categories: Close and Far for depth data. “Close” means that persons are close enough to begin to shake hands, and “Far” means that persons keep walking. The encoding mechanism is determined by the minimal distance extracted within the ROI. If the minimal distance is less than a threshold, that is, 50 cm in our experiments, the category is “Close” and vice versa. The reserved bits are designed for the additional sensor information in the future work.

3.2. Results Analysis. We chose the first five sets of training data in each group to swarm the optimal HTM network parameters. The final swarming results of main parameters for SP and TM algorithms described in the previous section are listed in Table 2. With these optimal parameters and rest of training data, we examined the skill learning performance by offline and online form, respectively. Offline validation is a paradigm of batch testing; that is, the skeleton and depth data collected by camera 1 (perception camera) at sampling time t_i ($i = 1, \dots, n - 1$) were first encoded as the sequential binary strings and then sent to the HTM network to obtain a batch of one-step ahead predicted action skeleton data for sampling time t_j ($j = 2, \dots, n$). We transferred the predicted skeleton to the joint coordinate system of NAO simulator and NAO simulator retrieved the shaking hands in batch form. The predictions are compared with the original skeleton data recorded by camera 2 (action camera) at t_j ($j = 2, \dots, n$), and the compared joints trajectories and statistical results are shown in Figure 7 and Table 3, respectively. Since in Case 1 NAO simulator stood statically and only shook right hand, the joints data of right arm are recorded only in Figure 7(a), where the robot completed the task within

20th to 100th sampling time. Figures 7(b)–7(e) illustrate all joints trajectories, where robot shakes hand within 100th to 180th sampling time and it walks 0.5 meters from 0 to 100th sampling time. It can also be found in Figure 7 that the predicted skeletons are consistent with the practical skeleton actions captured by camera 2. Table 3 shows the statistics for action predictions compared with the training data. It can be seen that the mean and variance for each prediction are close to zero, which manifests that the actions are predicted correctly and successfully. Figure 8 shows grabbed frames (the complete video clip can be found in attached media 1), where the left columns are for Case 1 and the right column is for Case 2. These offline examination results demonstrate that our proposed perception-action integration provides the correct action predictions according to the different perceived input data.

In online evaluation, a person stood in front of the RGB-D camera (refer to the camera in Figure 5). When the camera captures the person’s skeleton, the HTM network predicts the corresponding actions and then the actions are transferred to the joint positions for NAO simulator so that the NAO simulator can shake hands with the person interactively. Figure 9 displays the shaking hands interaction between NAO simulator and the person (the complete video clip can be found in attached media 2), where grabbed frames in the left column are for Case 1 and the right column is for Case 2. The joints trajectories of NAO simulator are shown in Figure 10. In comparison with the training data curves in Figure 7, it can be seen that the shapes of predicted skeleton data curves are similar to those of training data, which manifests that our proposed approach can also be used for online skill learning. In comparison with the trajectories in offline examination, it should be noted that in online evaluation the NAO simulator has a default initial motion of which the related trajectories are the data sampled from 0 to 50th time in Figures 10(a)–10(e). We do not consider these parts of joint trajectory in our proposed skill learning framework and just simulated the initial actions of the real NAO robot. Additionally, the learned action skeleton data in the training process are remembered in the HTM, and they are treated as the reference for the predicted actions. If the prediction is abnormal, these stored actions can be used for anomaly detection, which is discussed in Section 4.

The computational platform is a Corel i7-6500U 2.50 GHz, with a 12 G RAM laptop. The time for swarming optimal parameters of HTM network is around 60 minutes (the number of training data lines is around 1000). The online evaluation process, which consists of loading optimal parameters of HTM network, grabbing a frame of skeleton and depth data, encoding these perception data, implementing SP and TM algorithms, and output predictions, consumes 2.35 seconds. The cost time of online validation is considerably less than that of the training because the training is an optimal searching processing which is usually time-consuming. Additionally, only one frame of RGB-D data has to be processed; hence, the computational time is reduced considerably. Considering the results in terms of time cost, it is reasonable to use the proposed perception-action integration for real-time skill learning tasks.

TABLE 2: Main optimal parameters for SP and TM algorithms.

Parameters	Description	Value
Th_{seg}	Threshold for the number of active synapses on a segment	15
Th_{per}	Threshold for the permanence of potential synapse	0.2
bf_{ini}	Initial value of the boost factor	1.0
bf_{max}	Maximal boost factor	2.0
$r_{inh,ini}$	Initial value of the inhibition radius	0
LA_{min}	Minimal number of winning columns	1
$pm_{syn,inc}$	Incremented permanence value in spatial pooling	0.05
$pm_{syn,dec}$	Decremental permanence value in spatial pooling	0.05
$Th_{syn,per}$	Any synapse whose permanence value is above this threshold will become an active synapse	0.1
ADC_{min}	Minimum active duty cycle	0.001
Th_{act}	Threshold used to determine whether a distal segment is activated	14
pm_{inc}	Incremented permanence value in temporal pooling	0.1
pm_{dec}	Decremental permanence value in temporal pooling	0.1

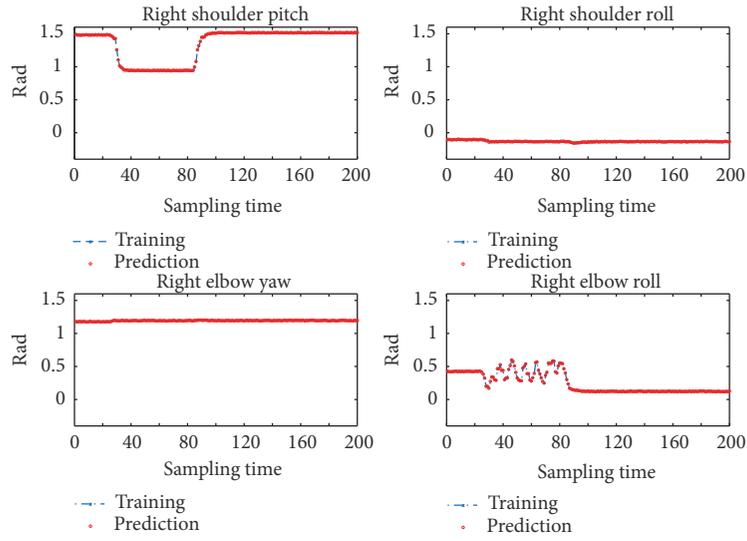
TABLE 3: Statistics for offline evaluation.

Case	LShoulder Pitch		LShoulder Roll		LElbow Yaw		LElbow Roll		LWrist Yaw	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0051	$0.819e-5$	-0.0015	$0.068e-5$	-0.0048	$0.786e-5$	-0.0049	$0.814e-5$	-0.0015	$0.073e-5$
Case	LHip Roll		LHip Pitch		LKnee Pitch		LAnkle Pitch		LAnkle Roll	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0016	$0.078e-5$	-0.0052	$0.81e-5$	-0.0050	$0.827e-5$	-0.0051	$0.814e-5$	-0.0014	$0.075e-5$
Case	RShoulder Pitch		RShoulder Roll		RElbow Yaw		RElbow Roll		RWrist Yaw	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	-0.0051	$0.822e-5$	-0.0048	$0.814e-5$	-0.0050	$0.775e-5$	-0.0048	$0.906e-5$	—	—
2	-0.0048	$0.862e-5$	-0.0015	$0.076e-5$	-0.0050	$0.86e-5$	-0.0051	$0.824e-5$	-0.0015	$0.081e-5$
Case	RHip Roll		RHip Pitch		RKnee Pitch		Rankle Pitch		Rankle Roll	
	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2	μ	σ^2
1	—	—	—	—	—	—	—	—	—	—
2	-0.0015	$0.072e-5$	-0.0051	$0.797e-5$	-0.0049	$0.881e-5$	-0.0049	$0.893e-5$	-0.0016	$0.077e-5$

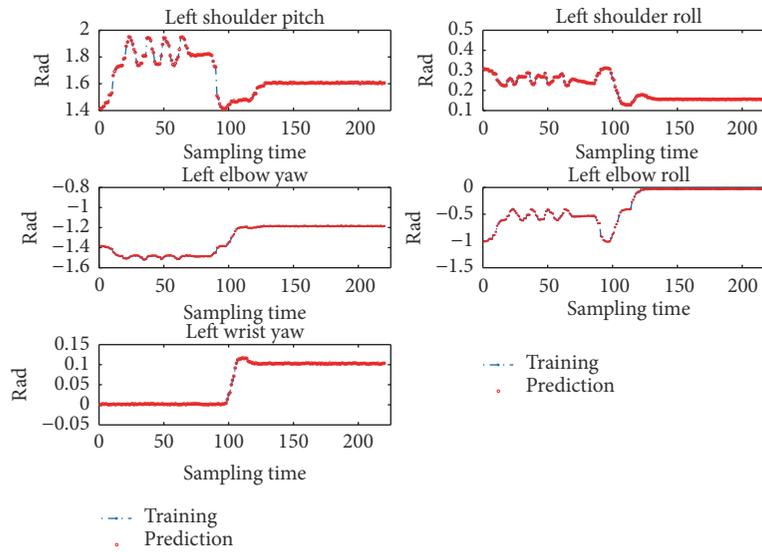
4. Discussion

4.1. Anomaly Detection. There is an important issue to be considered in the online evaluation. If the predicted actions deviate from those expected, the robot likely fails in the tasks of shaking hands. This situation is referred to in the terms of NuPIC as an *anomaly*. It is valuable to detect anomalies in real-time for many applications. CLA takes the *anomaly likelihood* computed from an anomaly score, a powerful anomaly detection analysis approach, to address this problem [24]. The anomaly likelihood enables the CLA to provide a metric representing the degree to which each record of the input sequence is predictable. It is relative to the data stream rather than an absolute measurement of abnormal behavior and is thus a critical reference to detect whether the pattern with a high anomaly score is actually

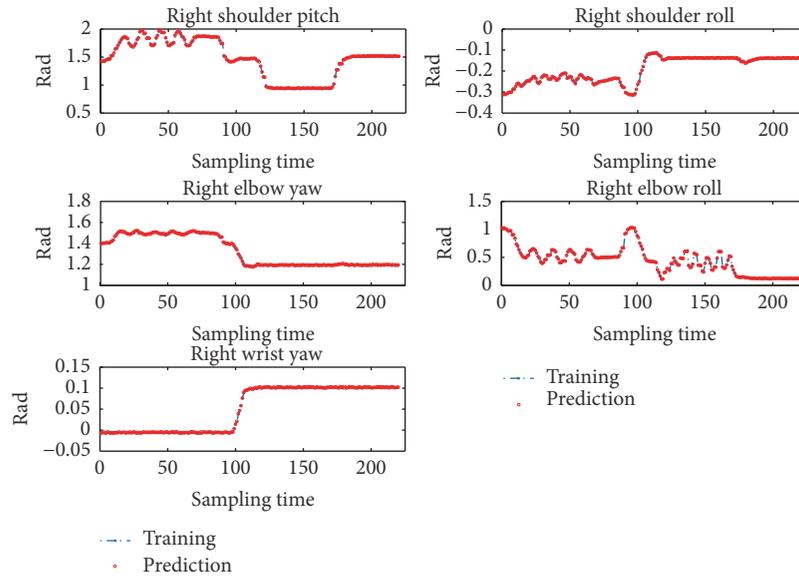
anomalous. Anomaly likelihood creates an average of the error score and then compares the current average error to a distribution of what the average error has been over the past data stream. This allows us to identify anomalies based on probability. As shown in Figure 11, if the anomaly likelihood is in the green section, this suggests that the record is normal. If it is in the red section, the record shows an abnormal value, which indicates that the pattern is a novel one not seen in any sequence. The yellow section indicates the pattern is somewhat unusual and that we do not have high confidence. In our application, we consider a pattern anomalous if its likelihood is in the yellow section. Based on the concept of anomaly detection, we calculated the anomaly likelihood for each predicted action in the shaking hands learning task. If the anomaly likelihood of any action is above a predefined probability threshold $P_{Th,ano}$ (0.90 in



(a) Joints trajectories in Case 1

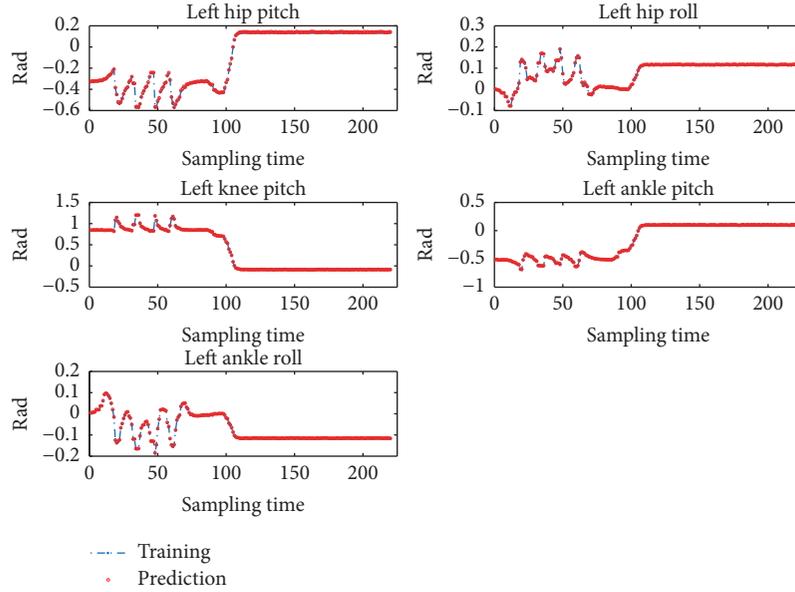


(b) Joints trajectories in Case 2

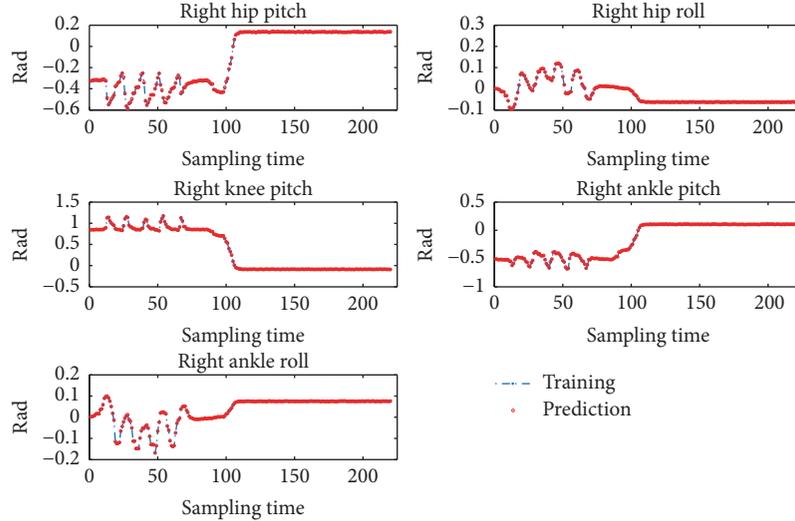


(c) Joints trajectories in Case 2 (cont.)

FIGURE 7: Continued.



(d) Joints trajectories in Case 2 (cont.)



(e) Joints trajectories in Case 2 (cont.)

FIGURE 7: Joints trajectories compared with the training data in offline evaluation.

our experiment, that is, the probability or accuracy of the green section is 90%, which is equivalent to a 1.65σ tolerance interval for a normal distribution), we designed a simple action retrieval strategy, that is, recalling the stored cells' active distal dendritic segments corresponding to the action sequence of training data to replace the predicted action which has a higher anomaly likelihood. The retrieved action is treated as the prediction for the next time.

It can be found in Figure 8 that the person's hand skeleton data in the grabbed frame of Case 1 video clip are deviation. The anomaly likelihood of predicted action corresponding to this perceived hand skeleton is 0.954, which is over 0.90. We replaced this anomalous action with the stored action during the training and sent it back to HTM as the prediction for

the next time. With this replacement procedure, the following predicted actions were correctly maintained and the NAO simulator continued to shake hands correctly. Because the CLA prediction mechanism in our experiment is one step ahead, we only retrieved one predicted action. If a multistep ahead prediction mechanism is adopted, the number of action retrievals is determined by the number of prediction steps and anomaly likelihoods.

4.2. Biological Evidence for Action Prediction. Learning the incorporated actions from different persons is an important cognitive function in the perception-action integration system, which has been examined by Knoblich and Flach [25]. They also proved that this type of prediction becomes more



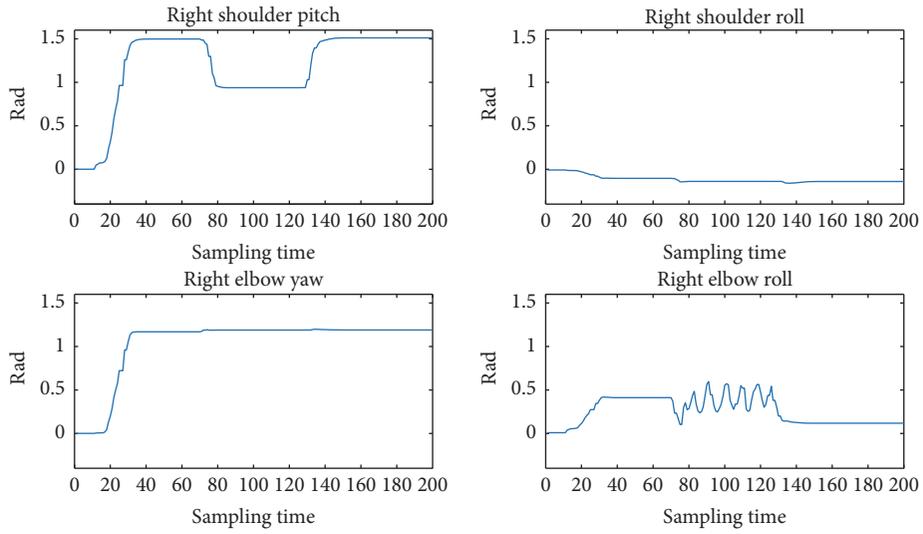
FIGURE 8: Grabbed frames of shaking hands learning in offline evaluation.



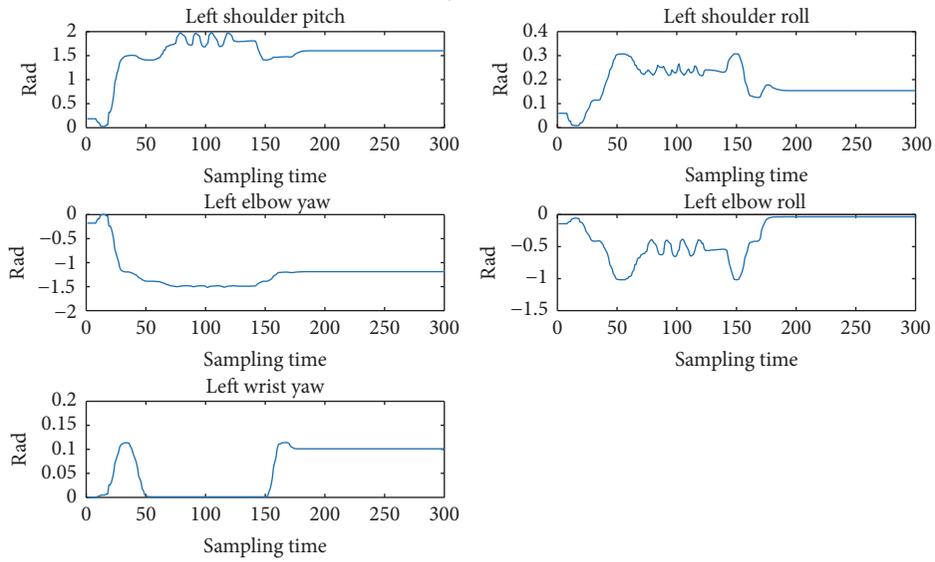
FIGURE 9: Grabbed frames of shaking hands learning in online evaluation.

accurate when one obtains the knowledge from one's own actions rather than those of others. Their research provides the biological evidence to support the action prediction mechanism of HTM and its application for skill learning tasks. However, the current HTM only implements a simple consequence prediction. It provides a sequence of predicted actions, including one-step or multistep predictions, but does not consider the potential information behind these

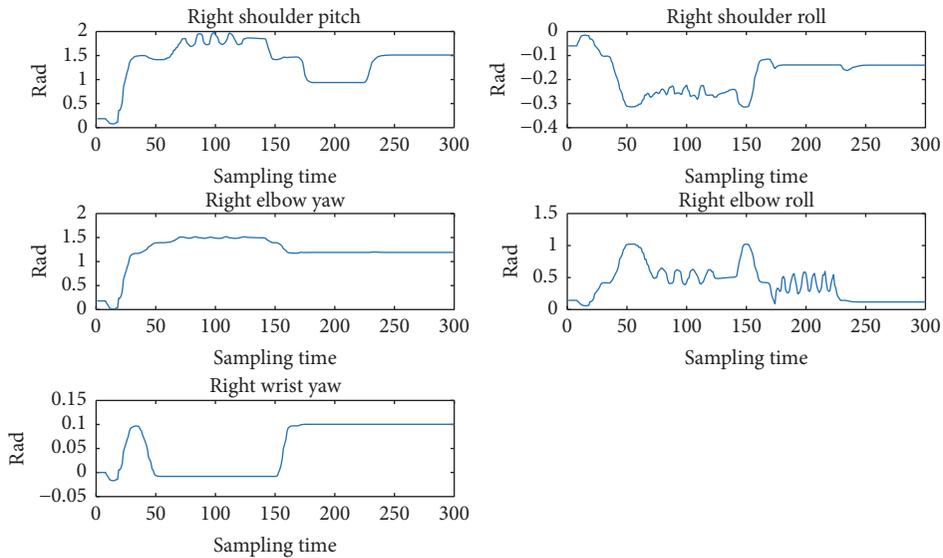
predictions. From a biological viewpoint, the present version of HTM does not link the perceptual input with the action system to predict the future outcome of actions [25]; that is, it does not explain the perception of intentionality for goal-related actions [26] or implement the understanding of the intention hidden in the sequential predicted actions [27] and how to learn to perceive something new [28]. Additionally, how the predicted actions guide the future perception process



(a) Joints trajectories in Case 1

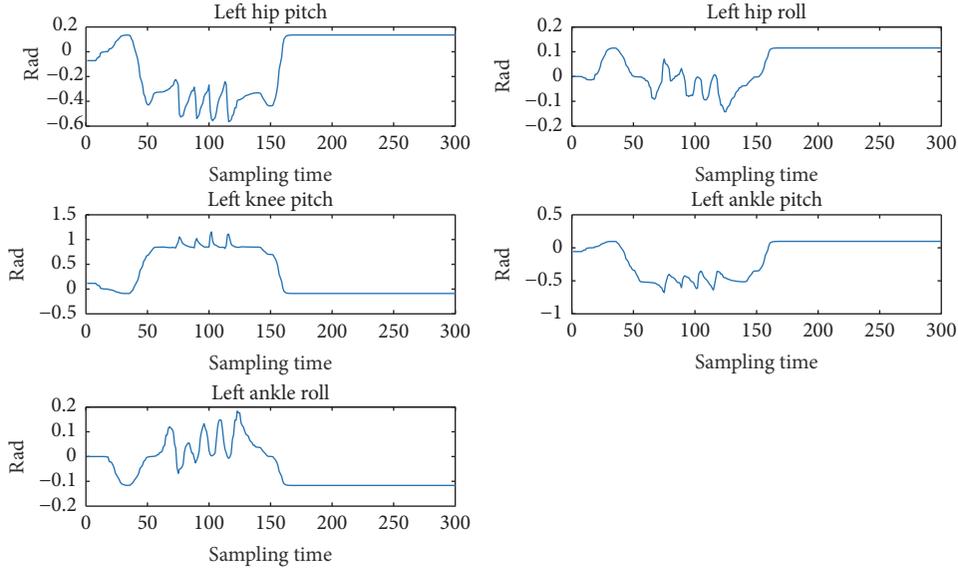


(b) Joints trajectories in Case 2

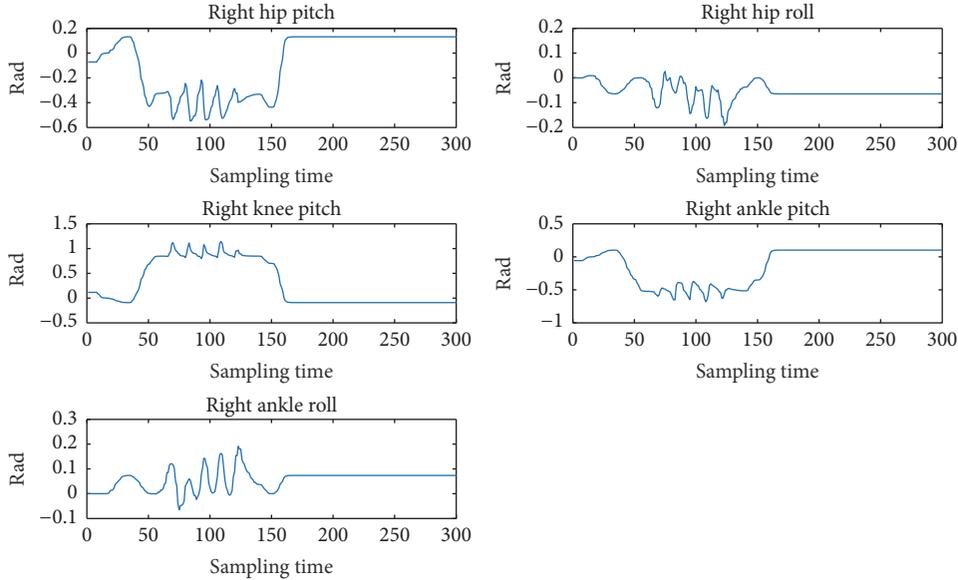


(c) Joints trajectories in Case 2 (cont.)

FIGURE 10: Continued.



(d) Joints trajectories in Case 2 (cont.)



(e) Joints trajectories in Case 2 (cont.)

FIGURE 10: Joints trajectories in online evaluation.

is not considered. Therefore, both of these two issues above will be the topics of our future work.

5. Conclusion

This study is the first attempt to explore the perception-action integration from the view of HTM for skill learning issue in intelligent robot. The main concept is that sequential perceptual information contributes to predicting one-step future actions. We selected the shaking hands as an example to evaluate the skill learning performance of our proposed framework on the NAO simulator. The perceived skeleton

of the target person and depth data from the target person are grabbed from a RGB-D camera. The perception data are first encoded as a sequential binary string. By using the SP algorithm, the binary string sequences are organized as a 2D SDRs. With the SDRs, TM algorithm makes predicted skeleton data for NAO simulator via storing the transitions between the current perceived skeleton data and predictions for the next future time. The prediction data are transformed to the joint coordinates framework so that the NAO simulator can implement the hands shaking actions with a real person. The experimental results manifest that the proposed method in this paper is promising for the skill learning of intelligent robots.

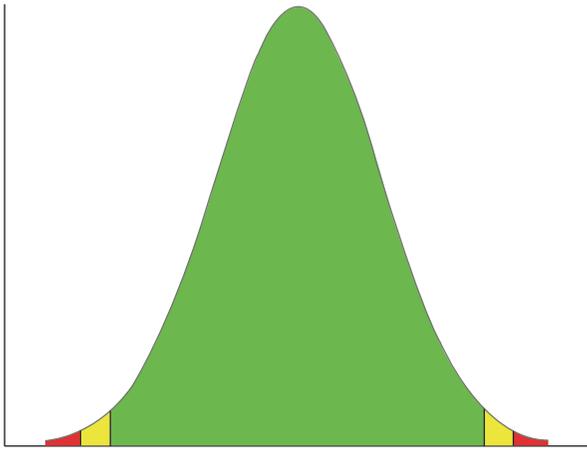


FIGURE 11: Anomaly likelihood curve.

Conflicts of Interest

The authors declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant no. 61203338. The authors thank the NuPIC Open Source Project and all the contributors of the NuPIC codes.

References

- [1] O. Sigaud and J. Peters, *From Motor Learning to Interaction Learning in Robots*, Springer, Berlin, Germany, 2010.
- [2] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in Cognitive Sciences*, vol. 3, no. 6, pp. 233–242, 1999.
- [3] J. M. Fuster, "Upper processing stages of the perception-action cycle," *Trends in Cognitive Sciences*, vol. 8, no. 4, pp. 143–145, 2004.
- [4] N. L. Cassimatis, J. G. Trafton, M. D. Bugajska, and A. C. Schultz, "Integrating cognition, perception and action through mental simulation in robots," *Robotics and Autonomous Systems*, vol. 49, no. 1-2, pp. 13–23, 2004.
- [5] J.-L. Schwartz, A. Basirat, L. Ménard, and M. Sato, "The Perception-for-Action-Control Theory (PACT): A perceptuo-motor theory of speech perception," *Journal of Neurolinguistics*, vol. 25, no. 5, pp. 336–354, 2012.
- [6] D. M. Wolpert, J. Diedrichsen, and J. R. Flanagan, "Principles of sensorimotor learning," *Nature Reviews Neuroscience*, vol. 12, no. 12, pp. 739–751, 2011.
- [7] K. P. Körding and D. M. Wolpert, "Bayesian integration in sensorimotor learning," *Nature*, vol. 427, no. 6971, pp. 244–247, 2004.
- [8] J. Piater, S. Jodogne, R. Detry et al., "Learning visual representations for perception-action systems," *International Journal of Robotics Research*, vol. 30, no. 3, pp. 294–307, 2011.
- [9] M. Do, J. Schill, J. Ernesti, and T. Asfour, "Learn to wipe: a case study of structural bootstrapping from sensorimotor experience," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '14)*, pp. 1858–1864, Hong Kong, May–June 2014.
- [10] H. Guedjou, S. Boucenna, and M. Chetouani, "Posture recognition analysis during human-robot imitation learning," in *Proceedings of the Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob '16)*, pp. 193–194, Cergy-Pontoise, France, September 2016.
- [11] A. Sadeghipour and S. Kopp, "Embodied gesture processing: motor-based integration of perception and action in social artificial agents," *Cognitive Computation*, vol. 3, no. 3, pp. 419–435, 2011.
- [12] J. Zhong, A. Cangelosi, and S. Wermter, "Toward a self-organizing pre-symbolic neural model representing sensorimotor primitives," *Frontiers in Behavioral Neuroscience*, vol. 8, article 22, 2014.
- [13] V. Cutsuridis and J. G. Taylor, "A cognitive control architecture for the perception-action cycle in robots and agents," *Cognitive Computation*, vol. 5, no. 3, pp. 383–395, 2013.
- [14] T. Shi, W. Yang, and H. Ren, "A study on the cognitive model of robot sensorimotor system," *Journal of Intelligent and Fuzzy Systems*, vol. 28, no. 5, pp. 1955–1968, 2015.
- [15] R. Kurzweil, *How to Create a Mind: The Secret of Human thought Revealed*, Viking, 2013.
- [16] J. Hawkins, S. Ahmad, S. Purdy, and A. Lavin, *Biological and Machine Intelligence (BAMI)*, 2016, <http://numenta.com/biological-and-machine-intelligence/>.
- [17] J. Hawkins, *On Intelligence*, Times Books, New York, NY, USA, 2004.
- [18] D. Huber, D. A. Gutnisky, S. Peron et al., "Multiple dynamic representations in the motor cortex during sensorimotor learning," *Nature*, vol. 484, no. 7395, pp. 473–478, 2012.
- [19] C. Lalanne and J. Lorenceau, "Crossmodal integration for perception and action," *Journal of Physiology—Paris*, vol. 98, no. 1-3, pp. 265–279, 2004.
- [20] E. Wieser and G. Cheng, "Progressive learning of sensory-motor maps through spatiotemporal predictors," in *Proceedings of the Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob '16)*, pp. 43–48, Cergy-Pontoise, France, September 2016.
- [21] Y.-S. Huang and Y.-J. Wang, "A hierarchical temporal memory based hand posture recognition method," *IAENG International Journal of Computer Science (IJCS)*, vol. 40, no. 2, pp. 87–93, 2013.
- [22] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '15)*, pp. 1110–1118, June 2015.
- [23] D. Rozado, F. B. Rodriguez, and P. Varona, "Extending the bioinspired hierarchical temporal memory paradigm for sign language recognition," *Neurocomputing*, vol. 79, pp. 75–86, 2012.
- [24] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, 2017.
- [25] G. Knoblich and R. Flach, "Predicting the effects of actions: interactions of perception and action," *Psychological Science*, vol. 12, no. 6, pp. 467–472, 2001.
- [26] A. E. Monroe, G. D. Reeder, and L. James, "Perceptions of intentionality for goal-related action: Behavioral description matters," *PLoS ONE*, vol. 10, no. 3, Article ID e0119841, 2015.

- [27] S.-J. Blakemore and J. Decety, "From the perception of action to the understanding of intention," *Nature Reviews Neuroscience*, vol. 2, no. 8, pp. 561–567, 2001.
- [28] E. A. Di Paolo, X. E. Barandiaran, M. Beaton, and T. Buhrmann, "Learning to perceive in the sensorimotor approach: Piaget's theory of equilibration interpreted dynamically," *Frontiers in Human Neuroscience*, vol. 8, article 551, 2014.

Research Article

Adaptive Neural Network Control of Serial Variable Stiffness Actuators

Zhao Guo,¹ Yongping Pan,^{2,3} Tairen Sun,⁴ Yubing Zhang,¹ and Xiaohui Xiao¹

¹School of Power and Mechanical Engineering, Wuhan University, Wuhan 430072, China

²Department of Biomedical Engineering, National University of Singapore, Singapore

³National University of Singapore (Suzhou) Research Institute, Suzhou 215123, China

⁴Department of Electrical and Information Engineering, Jiangsu University, Zhenjiang 212013, China

Correspondence should be addressed to Yongping Pan; yongppan@gmail.com

Received 13 August 2017; Revised 8 September 2017; Accepted 20 September 2017; Published 8 November 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Zhao Guo et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper focuses on modeling and control of a class of serial variable stiffness actuators (SVSAs) based on level mechanisms for robotic applications. A multi-input multi-output complex nonlinear dynamic model is derived to fully describe SVSAs and the relative degree of the model is determined accordingly. Due to nonlinearity, high coupling, and parametric uncertainty of SVSAs, a neural network-based adaptive control strategy based on feedback linearization is proposed to handle system uncertainties. The feasibility of the proposed approach for position and stiffness tracking of SVSAs is verified by simulation results.

1. Introduction

Variable stiffness actuators (VSAs), which can increase safety in physical human-robot interaction and meet the dynamic requirements of robots in unknown environments, have been developed for a new generation of robots [1–11]. VSAs present advantages in terms of mechanical stiffness adjusting, energy storing, and force sensing capabilities [3]. According to the design configurations, there are two typical mechanical arrangements for VSAs: one is the antagonistic configuration with a pair of actuators coupled via nonlinear springs [10], and the other is the serial configuration with two independent motors, where a principal motor drives the joint position through a compliant transmission, and a secondary motor adjusts the stiffness [11].

Antagonistic-type VSAs need two driving units to change the stiffness and position synchronously, which lead to high energy consumption and complex control design. Serial-type VSAs (SVSAs) do not have this requirement and have gained more attention. Recently, new types of SVSAs have been developed based on level arm mechanisms, where the mechanical stiffness is regulated by moving one of the level points: a pivot point, a spring located point, or a

force point [12–14]. Thus, the SVSAs are not significantly influenced by the coupling between the load and the stiffness transmission mechanism. It also brings higher energy efficiency in unloaded conditions [15]. Based on the variable ratio lever principle by changing the pivot position, a novel compact rotational SVSA with an Archimedean spiral relocation mechanism (ASRM) was developed to obtain compact mechanical structure, large adjustable stiffness range, and better force transmission ability in [15]. In this innovative design, the ASRM is applied to transfer the rotation of the Archimedean spiral cam into the linear motion of the pivot, and an elastic force transmission mechanism with a spring shaft vertical to the output link is proposed to achieve large deflection angle and high energy storage ability.

VSAs are multi-input multioutput (MIMO), highly coupled, and complex nonlinear systems which include both structured and unstructured uncertainties [16]. In VSAs, the stiffness variation brings physical modifications, requiring the control system to quickly transit among different operating conditions. In addition, the coupling between stiffness and positioning mechanisms and the increased system dimensionality complicate the entire control system [16]. As a result, it is challenging to design control strategies for VSAs

in robotic applications. Different control strategies have been proposed to for VSAs, where the simplest one is the PD control in [16]. However, the performance of the PD control highly depends on PD gains due to the high nonlinearity of VSAs. In [17, 18], a classical nonlinear control technique termed feedback linearization was explored for VSAs. Nevertheless, feedback linearization is very sensitive to modeling accuracy so that significant efforts in system modeling and parameter identification are required to achieve a desired performance. Based on the feedback linearization, some advanced control methods, such as gain-scheduling control [19], active damping control [20], backstepping control [21], nonlinear model predictive control [22], and output feedback control [23], have been integrated to improve stiffness and position tracking performances of VSAs. Although these control methods are proved to be effective for VSAs, they may exhibit undesired performances especially if the VSA dynamics are highly nonlinear and experience large parameter variations due to the changes of the actuator stiffness and load conditions [24].

Neural networks (NNs) have been widely applied to control complex nonlinear systems, especially complex robotic systems in recent years [25–37]. Compared with the traditional adaptive control, NN adaptive control (NNAC) has two prominent attractive features [38]. Firstly, due to the universal approximation property of NNs, the difficulty of system modeling in many practical control problems can be largely alleviated. Secondly, due to the practical persistently exciting (PE) condition of NNs, parameter convergence is easier to be obtained during control resulting in an exact online modeling and superior exponential tracking. Although NNAC is promising for complex nonlinear systems like VSAs, it is seldom applied to VSAs except [39], where antagonistic VSAs are considered in [39].

This paper focuses on modeling and control of a class of SVSAs based on level mechanisms. Firstly, a MIMO complex nonlinear dynamic model is derived to fully describe SVSAs and the relative degree of the model is determined accordingly; secondly, based on the feedback linearization, a direct NNAC strategy is proposed to govern the SVSA model; finally, high-fidelity simulations are provided to verify the effectiveness of the proposed approach. Compared with other control strategies for VSAs, the proposed NNAC can deal with high coupling and parametric uncertainties of SVSAs. In addition, the control on both position and stiffness tracking remain effective under load changing conditions.

2. Serial Variable Stiffness Actuator Modeling

As illustrated in Figure 1, a compact rotational SVSA consists of a variable stiffness mechanism (VSM), a principal motor, and a secondary motor, where the principal motor drives the output link motion through the spring transmission, and a secondary motor adjusts the stiffness of the actuator by changing the position of the pivot with an ASRM [15].

By considering gravity and external loads, the dynamic model of SVSAs is represented as follows:

$$\begin{aligned} M\ddot{q} + D\dot{q} + \tau_e(\theta_2, \varphi) + \tau_g(q) &= \tau_{\text{ext}} \\ B_1\ddot{\theta}_1 + D_1\dot{\theta}_1 - \tau_e(\theta_2, \varphi) &= u_1 \\ B_2\ddot{\theta}_2 + D_2\dot{\theta}_2 + \tau_r(\theta_2, \varphi) &= u_2, \end{aligned} \quad (1)$$

where q is a position of the output link, θ_i with $i = 1, 2$ is a position associated with each motor, $\varphi := q - \theta_1$ is a deflection angle of the elastic transmission, M is an inertia of the output link, B_i is a reflected inertia of each motor, D is a reflected damping of the link, D_i is a reflected damping of each motor, $\tau_g(q)$ is a gravity torque, u_i is a control input of each motor, τ_{ext} is an external torque, τ_r is a coupling reaction torque, and τ_e is an elastic torque of the spring transmission.

The elastic torque across the transmission is given by

$$\tau_e = K_s R^2 \mu^2 \frac{\sin \varphi \cos \varphi}{(1 - \mu \cos \varphi)^2}, \quad (2)$$

where K_s is a spring stiffness, R is a radius of the output link, and μ is a lever length ratio. The coupled flexibility torque, representing the transmission deformation reacting on the secondary motor, is given by

$$\tau_r = K_s R^2 a^2 \frac{\sin 2\beta \sin^2 \varphi}{2(R - a \cos \varphi)(a^2 + R^2 - 2aR \cos \varphi)}, \quad (3)$$

where $\beta = \arctan(-\theta_2/\gamma)$ is a tangent angle of the Archimedean spiral gear, γ is a reduction gear ratio of the secondary motor, and $a = \mu R = R\theta_2/2\pi$ is a distance from the pivot point to the joint center. The stiffness of this SVSA is formulated by

$$\sigma(\theta_2, \varphi) = \frac{\partial \tau_e(\theta_2, \varphi)}{\partial \varphi} = K_s R^2 \mu^2 \frac{\cos 2\varphi - \mu \cos \varphi}{(1 - \mu \cos \varphi)^3}. \quad (4)$$

The level length ratio μ can be represented by the position of the secondary motor as follows:

$$\mu = \frac{\theta_2}{2\pi\gamma} + \mu_0, \quad (5)$$

where μ_0 is an initial level length ratio.

3. Neural Network Control Design

3.1. Problem Formulation. The dynamic model of SVSAs given by (1)–(5) can be represented in the standard form

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x}) \mathbf{u} \\ \mathbf{y} &= \mathbf{h}(\mathbf{x}), \end{aligned} \quad (6)$$

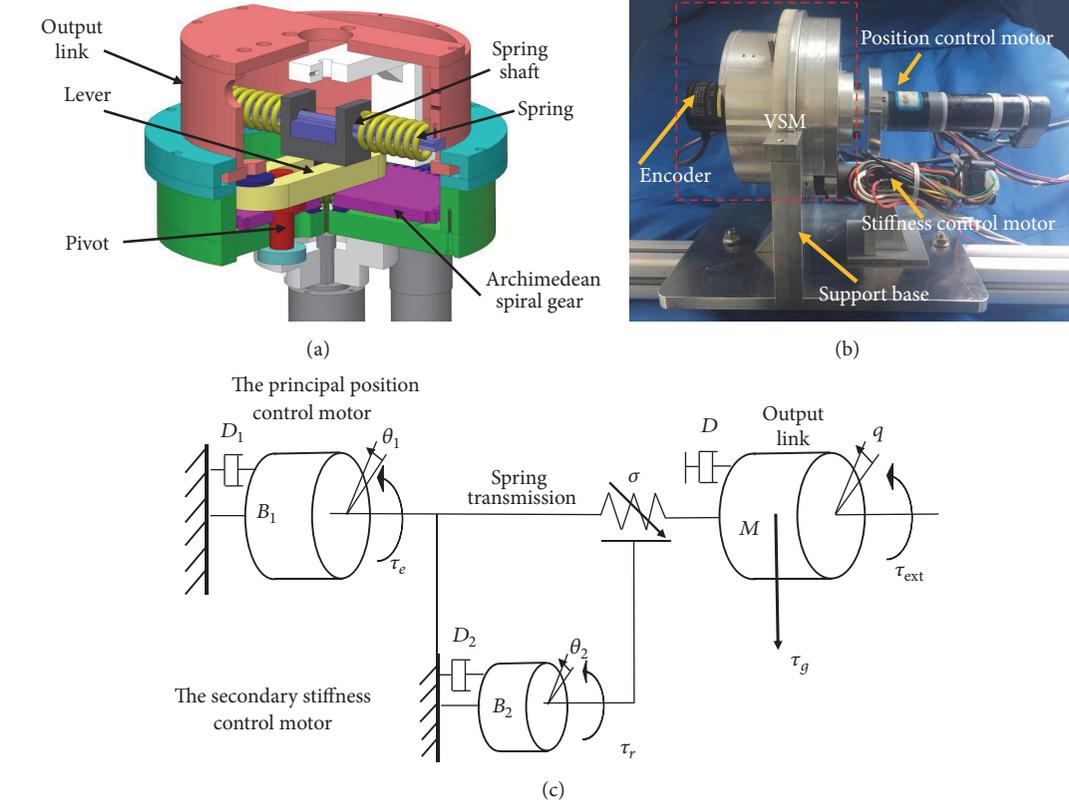


FIGURE 1: CAD model (a), prototype (b), and schematic model (c) of the SVSA.

where $\mathbf{x} = [q, \dot{q}, \theta_1, \dot{\theta}_1, \theta_2, \dot{\theta}_2]^T$ is a vector of system states, $\mathbf{u} = [u_1, u_2]^T$ is a control input, $y = [q, \sigma]^T$ is a system output, and

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \dot{q} \\ M^{-1}(-D\dot{q} - \tau_e + \tau_{\text{ext}} - \tau_g) \\ \dot{\theta}_1 \\ B_1^{-1}(-D_1\dot{\theta}_1 + \tau_e) \\ \dot{\theta}_2 \\ B_2^{-1}(-D_2\dot{\theta}_2 - \tau_r) \end{bmatrix},$$

$$\mathbf{g}(\mathbf{x}) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ B_1^{-1} & 0 \\ 0 & 0 \\ 0 & B_2^{-1} \end{bmatrix},$$

$$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} q \\ \sigma \end{bmatrix}.$$

The equilibrium point of system (6) is $\mathbf{x}_0 = 0$. Let $q_d \in R$ and $\sigma_d \in R$ be bounded desired outputs. Define tracking errors $e_q := q - q_d$ and $e_\sigma := \sigma - \sigma_d$.

Assumption 1. There exist uncertainties in $\mathbf{f}(\mathbf{x})$ and $\mathbf{g}(\mathbf{x})$ due to modeling difficulties and modeling errors.

Assumption 2. $\ddot{q}_d, \ddot{\sigma}_d, \ddot{q}_d, \ddot{\sigma}_d, \ddot{q}_d, \ddot{\sigma}_d \in R$ exist and are bounded.

The objective of this study is to design NNAC law without the exact knowledge of the SVSA model (1) such that the tracking errors e_q and e_σ are converging.

3.2. Feedback Linearization-Based Control

(7) *Definition 3* (the standard Lie derivative notation is used for the description [40]). The relative degree from the input \mathbf{u} to the output \mathbf{y} of system (6) at the equilibrium \mathbf{x}_0 is (σ_1, σ_2) if (a) $L_{g_j} L_f^k h_i(\mathbf{x}) = 0$ ($1 \leq j, i \leq 2$) for all $k < \sigma_i - 1$ and all \mathbf{x} in a neighborhood of \mathbf{x}_0 and (b) the matrix

$$\mathbf{G}(\mathbf{x}) = \begin{bmatrix} L_{g_1} L_f^{\sigma_1-1} h_1 & L_{g_2} L_f^{\sigma_1-1} h_1 \\ L_{g_1} L_f^{\sigma_2-1} h_2 & L_{g_2} L_f^{\sigma_2-1} h_2 \end{bmatrix} \quad (8)$$

is nonsingular at $\mathbf{x} = \mathbf{x}_0$.

By simple calculation, the input relative degree of system (6) is (4, 2) so that $G(\mathbf{x})$ can be rearranged as follows:

$$G(\mathbf{x}) = \begin{bmatrix} L_{g_1} L_f^3 h_1 & L_{g_2} L_f^3 h_1 \\ L_{g_1} L_f h_2 & L_{g_2} L_f h_2 \end{bmatrix}. \quad (9)$$

Therefore, there exists a diffeomorphism

$$\begin{bmatrix} q \\ \dot{q} \\ \ddot{q} \\ \sigma \\ \dot{\sigma} \end{bmatrix} = \begin{bmatrix} h_1(\mathbf{x}) \\ L_f h_1(\mathbf{x}) \\ L_f^2 h_1(\mathbf{x}) \\ L_f^3 h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \\ L_f h_2(\mathbf{x}) \end{bmatrix} \quad (10)$$

that transforms system (6) into

$$\begin{bmatrix} \dot{q}^{(4)} \\ \dot{\sigma}^{(2)} \end{bmatrix} = F(\mathbf{x}) + G(\mathbf{x}) \mathbf{u} \quad (11)$$

with $F: R^6 \mapsto R^2$ being a vector field.

Let $\mathbf{e}_q := [e_q, \dot{e}_q, \ddot{e}_q, \ddot{e}_q]^T$ and $\mathbf{e}_\sigma := [e_\sigma, \dot{e}_\sigma]^T$. Define

$$\begin{aligned} e_{s1} &= \left(\frac{d}{dt} + \lambda_1 \right)^3 e_q = [\lambda_1^3, 3\lambda_1^2, 3\lambda_1, 1] \mathbf{e}_q \\ e_{s2} &= \left(\frac{d}{dt} + \lambda_2 \right) e_\sigma = [\lambda_2, 1] \mathbf{e}_\sigma \end{aligned} \quad (12)$$

with $\lambda_1, \lambda_2 \in R^+$. Then, one gets

$$\dot{\mathbf{e}}_s = F(\mathbf{x}) + G(\mathbf{x}) \mathbf{u} + \mathbf{v} \quad (13)$$

with $\mathbf{e}_s := [e_{s1}, e_{s2}]^T$, $\mathbf{v} = [v_1, v_2]^T$, $v_1 := [0, \lambda_1^3, 3\lambda_1^2, 3\lambda_1] \mathbf{e}_q - q_d^{(4)}$, and $v_2 := [0, \lambda_2] \mathbf{e}_\sigma - \dot{\sigma}_d$. A feedback linearization-based ideal control law is given by

$$\mathbf{u} = G^{-1}(\mathbf{x}) (-K \mathbf{e}_s - F(\mathbf{x}) - \mathbf{v}) \quad (14)$$

with $K \in R^{2 \times 2}$ being a positive-definite diagonal matrix of control gains.

3.3. Adaptive Neural Network Control. As $F(\mathbf{x})$ in control law (14) is not exactly known, a radial basis function- (RBF-) NN represented as follows:

$$\hat{f}(\mathbf{x}, \widehat{W}) = \Phi^T(\mathbf{x}) \widehat{W} \quad (15)$$

is applied to approximate $F(\mathbf{x})$, where $\widehat{W} = [\widehat{W}_1, \widehat{W}_2] \in R^{N \times 2}$ with $\widehat{W}_i = [\widehat{w}_{i1}, \widehat{w}_{i2}, \dots, \widehat{w}_{iN}]^T$ ($i = 1, 2$) is a matrix of NN weights, $\Phi(\mathbf{x}) = [\phi_1(\mathbf{x}), \phi_2(\mathbf{x}), \dots, \phi_N(\mathbf{x})]^T \in R^N$ is a vector of regression functions, N is the number of neural nodes, and $\phi_j(\mathbf{x}): R^6 \mapsto R$ are commonly chosen to be Gaussian RBFs:

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_j\|^2}{(2r_j^2)}\right) \quad (16)$$

with $j = 1, 2, \dots, N$ and $\mathbf{c}_j = [c_{1j}, c_{2j}]^T$, where $c_{ij} \in R$ ($i = 1, 2$) and $r_j \in R^+$ are centers and widths of the Gaussian functions, respectively.

Let $\Omega_w := \{\widehat{W} \mid \|\widehat{W}\|_F \leq c_w\}$ and $\Omega_x := \{\mathbf{x} \mid \|\mathbf{x}\| \leq c_x\}$. Define an optimal NN approximation error

$$\varepsilon(\mathbf{x}) := f(\mathbf{x}) - \hat{f}(\mathbf{x}, W^*), \quad (17)$$

with a constant matrix of optimal weights

$$W^* := \arg \min_{\widehat{W} \in \Omega_w} \left\{ \sup_{\mathbf{x} \in \Omega_x} |f(\mathbf{x}) - \hat{f}(\mathbf{x}, \widehat{W})| \right\}. \quad (18)$$

According to the universal approximation property of RBF-NNs, given any small constant $\varepsilon^* \in \mathbb{R}^+$, there is a sufficiently large N so that $|\varepsilon(\mathbf{x})| \leq \varepsilon^*$, $\forall \mathbf{x} \in \Omega_x$ [41].

Assumption 4. There exists a matrix $G_0 = \text{diag}(g_{10}, g_{20})$ with $g_{10}, g_{20} \in R^+$ such that $G(\mathbf{x}) \geq G_0$ holds.

Now, the actual control law is designed as follows:

$$\mathbf{u} = G_0^{-1} (-K \mathbf{e}_s - \Phi^T(\mathbf{x}) \widehat{W} - \mathbf{v}), \quad (19)$$

and the update law of \widehat{W} is given by

$$\dot{\widehat{W}} = \Gamma \mathcal{P}(\Phi(\mathbf{x}) \mathbf{e}_s), \quad (20)$$

where $\Gamma \in R^{2 \times 2}$ is a positive-definite matrix of learning rates, and \mathcal{P} is a projection operator given by

$$\mathcal{P}(\bullet) = \begin{cases} \bullet, & \text{if } \|\widehat{W}\|_F < c_w \text{ or } \|\widehat{W}\|_F = c_w \text{ \& } \widehat{W}^T \bullet \leq 0 \\ \bullet - \frac{\widehat{W} \widehat{W}^T}{\|\widehat{W}\|_F^2} \bullet, & \text{otherwise.} \end{cases} \quad (21)$$

Note that the choice of the gain matrix G_0 is based on a rough estimation of the inertia matrices M , B_1 , and B_2 , where the exact values of M , B_1 , and B_2 are not required in practice. It follows from the standard NNAC result [41] that system (11) driven by control law (19) with (20) achieves practical asymptotic stability in the sense that e_q and e_σ converge to small neighborhoods of zero determined by K and ε^* .

4. Numerical Results

A general specification of SVSAs in Table 1 [15] is used for simulation. The construction of the proposed NNAC follows the following steps: (1) to construct regression functions $\phi_j(\mathbf{x})$ in (16), select three Gaussian functions to cover each universe of Ω_x such that $j = 1$ to 729; (2) set the filtered error parameters $\lambda_1 = \lambda_2 = 20$ in (12); (3) set the lower bound $G_0 = \text{diag}(50000, 1000)$ and the control gain $K = \text{diag}(20, 20)$ for the control law in (19); (4) set the learning rate matrix $\Gamma = \text{diag}(300, 100)$ for the adaptive law in (20).

A tracking task with the desired position $q_d(t) = 120 \sin(\pi t)$ (°) and the desired stiffness $\sigma_d(t) = 50 \sin(1.25\pi t)$

TABLE I: SVSA parameter specifications.

Description	Symbol (unit)	Value
Output link inertia	M (kg·m ²)	0.0063
Motor M1 + gearbox + intermediate inertia	B_1 (kg·m ²)	0.0134
Motor M2 + gearbox + ASRM inertia	B_2 (kg·m ²)	0.0110
Output link damping	D (N·ms/rad)	$1.6 * 10^{-3}$
Motor M1 damping	D_1 (N·ms/rad)	$3.0 * 10^{-3}$
Motor M2 damping	D_2 (N·ms/rad)	$2.4 * 10^{-3}$
Inherent spring stiffness	Ks (N/m)	3764.5
Radius (mm)	R (m)	0.05
Range of motion	(deg.)	0–360
Range of deflection angle	(deg.)	0–30
Range of stiffness	(N·m/rad)	1.7–150.5

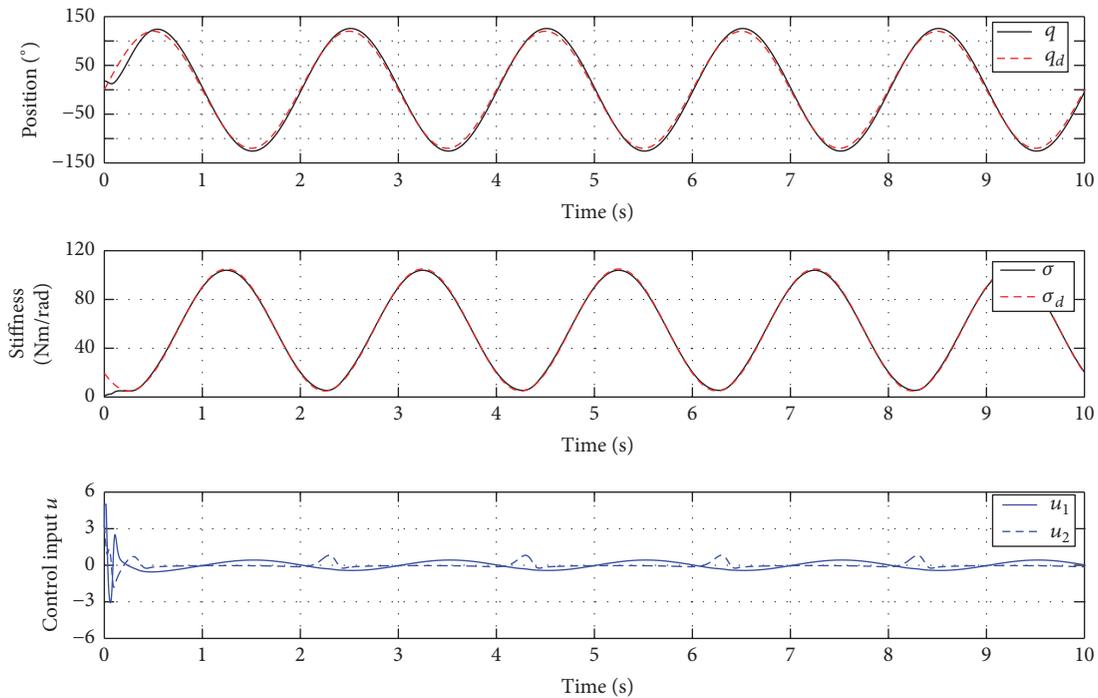


FIGURE 2: Control trajectories by the PD control without load.

$\pi t) + 55$ (Nm/rad) (frequency 0.5 Hz) is chosen to show the tracking performance of the proposed controller for SVSAs, where the desired stiffness σ_d ranges between 5 and 105 Nm/rad. To make comparisons, a PD controller is selected as a baseline controller, where PD gains are optimized to minimize errors under a reasonable control \mathbf{u} .

Tracking results by the PD control and the proposed NNAC are demonstrated in Figures 2 and 3, respectively, and comparisons of system errors are given in Figure 5, where no load is applied on the actuator in this case. It is shown that the proposed controller achieves a much higher position tracking accuracy than the PD control under a comparable control

input \mathbf{u} despite the change of the stiffness. The position tracking accuracies by the PD control and the proposed NNAC are about 0.4547° and 7.3490° , respectively. The stiffness tracking result also shows better performance of the proposed NNAC than the PD control, where the stiffness tracking accuracy reduces from 1.1200 Nm/rad to 0.2857 Nm/rad. In addition, Figure 4 shows that the norms of the NN weights \widehat{W}_1 and \widehat{W}_2 converge to some constants, which implies that an exact estimation of the system uncertainty is achieved according to the NN learning theory [38].

To investigate the adaptability of the proposed NNAC, a 2 kg load is added to the link at $t = 15$ s. System errors

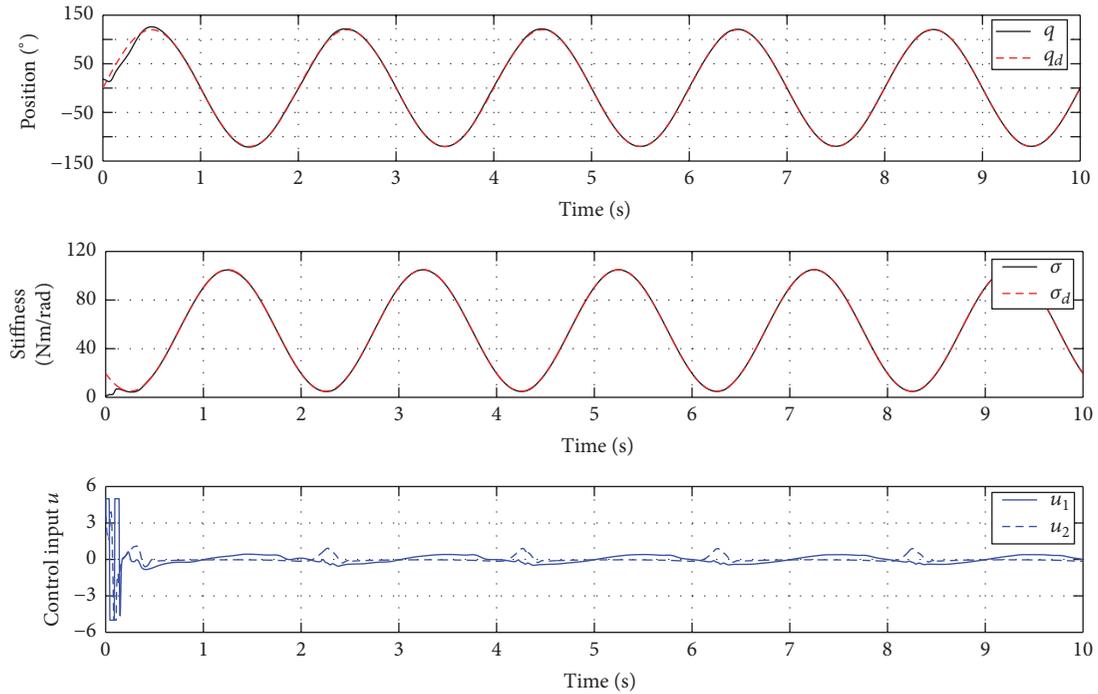


FIGURE 3: Control trajectories by the proposed NNAC without load.

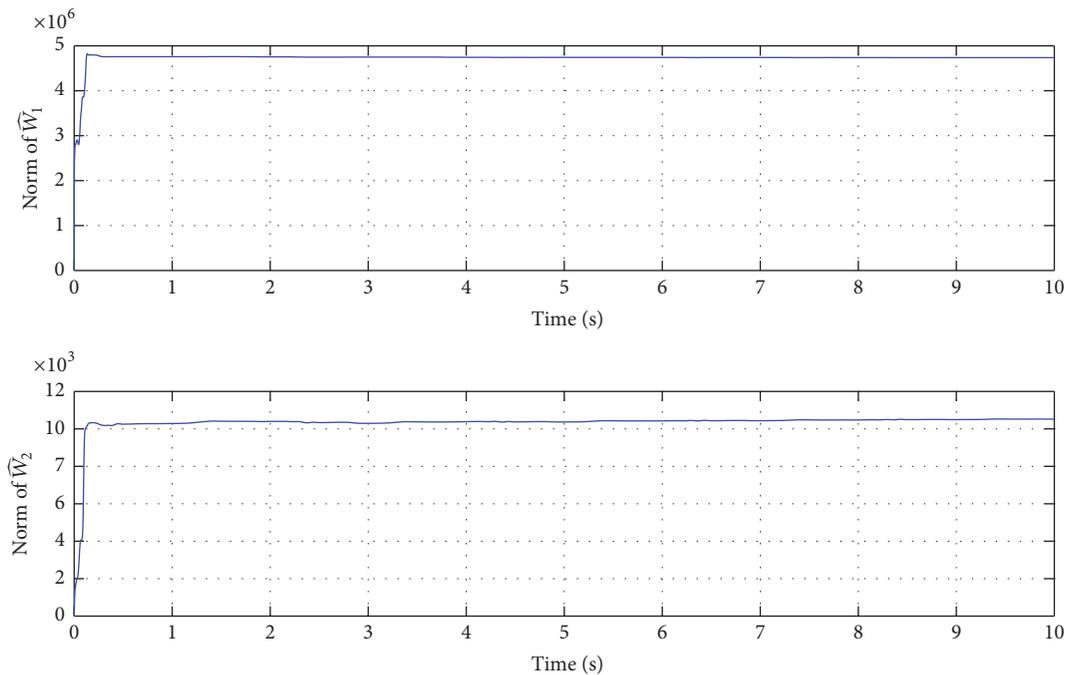


FIGURE 4: Evaluations of NN weights without load.

and control inputs within 30 s are depicted in Figure 6. It is shown that, at the time 15 s, both the position error e_q and the stiffness error e_σ increase dramatically but then reduce gradually to small values. After an adaptation about 15 s, the high position tracking accuracy is still maintained by the proposed NNAC, where the maximum position error e_q (0.8744°) is even smaller than that before adding the load (0.9265°). The adaptability of the proposed NNAC is apparent as

position error e_q continues decreasing to small range despite the unknown load added. Note that adding loads does not affect the stiffness tracking accuracy.

5. Conclusions

This paper has successfully applied a NNAC method to control SVSAs. Simulation results verify the ability of the

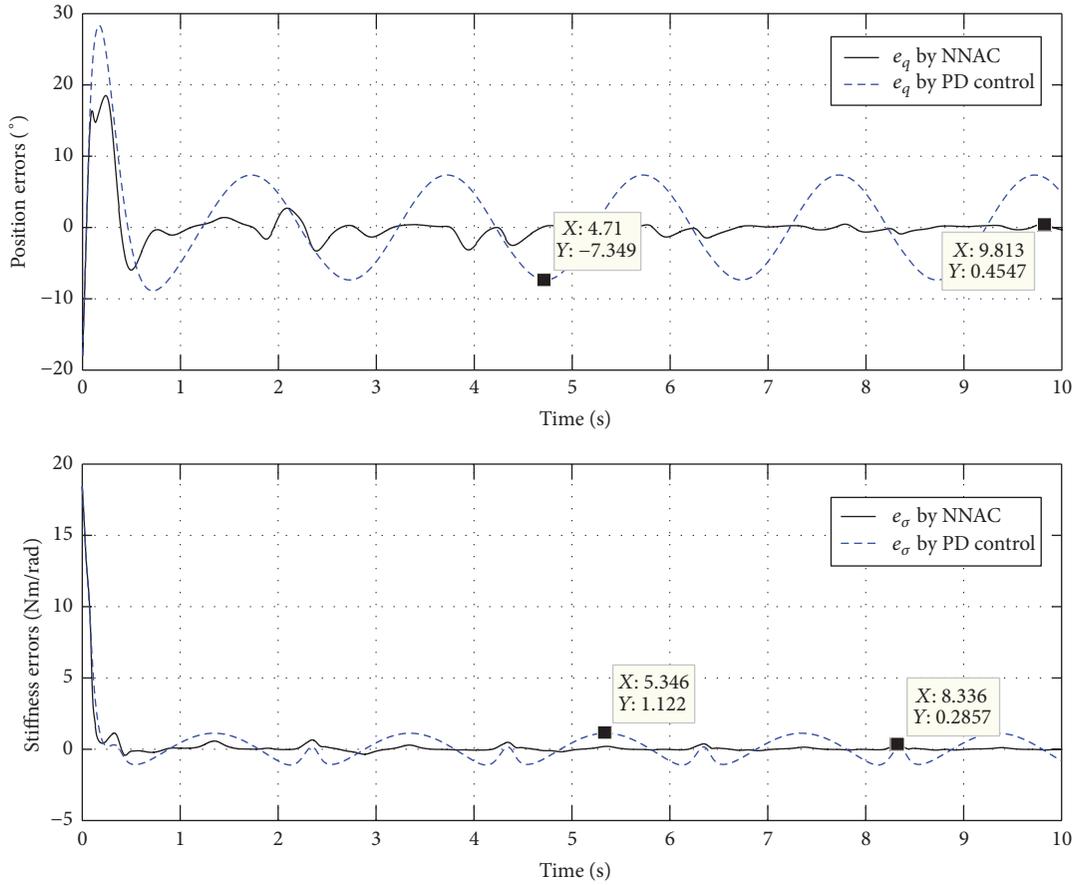


FIGURE 5: Comparisons of system errors without load.

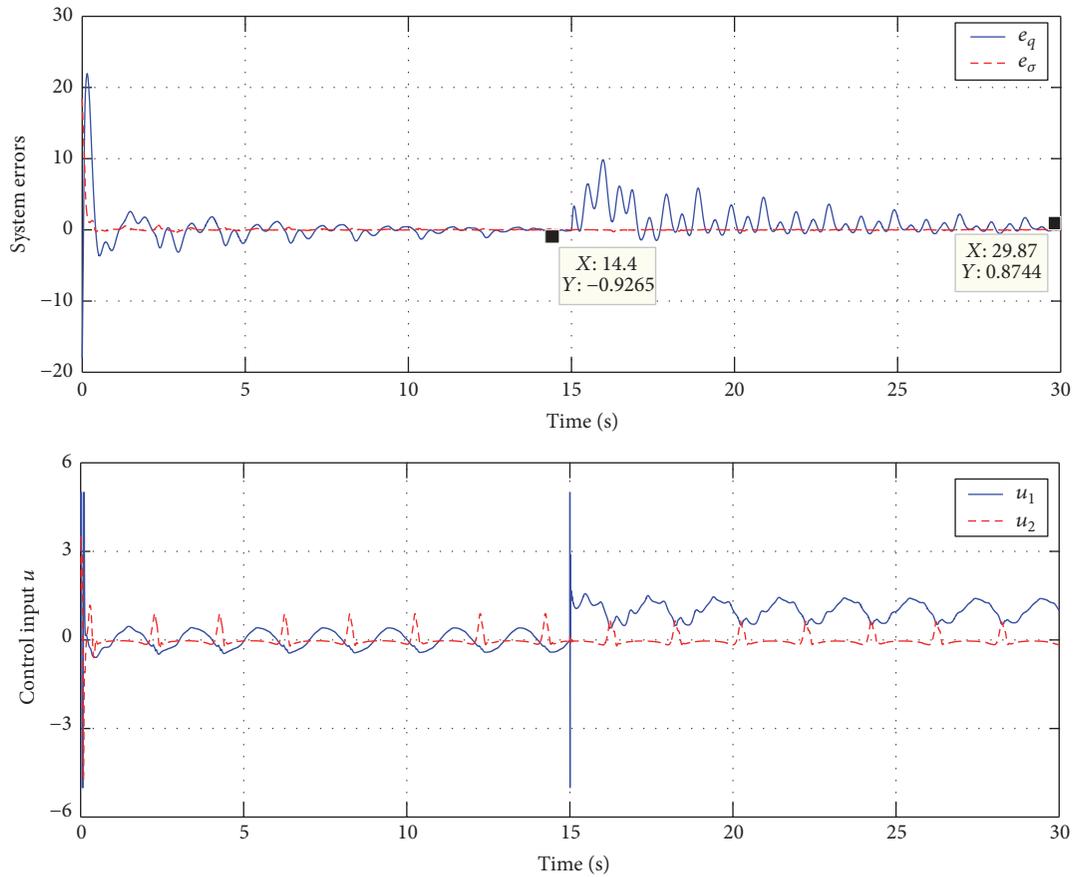


FIGURE 6: Control trajectories by the proposed NNAC with load.

proposed approach to cope with system variability, by showing remarkable control performances for both position and stiffness tracking under load changes. In future work, the practicability of the proposed NNAC would be enhanced by the composite learning [42] and the observer design [43], and experimental validity of the proposed controller will be carried out on a physical SVSA system.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (nos. 61703295, 51605339, and 51675385), the Natural Science Foundation of Jiangsu Province (no. BK20150385), the Natural Science Foundation of Hubei Province (no. 2017CFB496), the China Postdoctoral Science Foundation (nos. 2016M592382 and 2017T100573), Wuhan Youth Science and Technology Dawn Plan (no. 2017050304010304), and the Fundamental Research Funds for the Central Universities (no. 2042016kf0021).

References

- [1] B. Vanderborght, A. Albu-Schaeffer, A. Bicchi et al., "Variable impedance actuators: a review," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1601–1614, 2013.
- [2] S. Wolf and G. Hirzinger, "A new variable stiffness design: Matching requirements of the next robot generation," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation, ICRA 2008*, pp. 1741–1746, usa, May 2008.
- [3] S. Wolf, G. Grioli, O. Eiberger et al., "Variable Stiffness Actuators: Review on Design and Components," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 5, pp. 2418–2430, 2016.
- [4] H. Yu, S. Huang, G. Chen, and N. Thakor, "Control design of a novel compliant actuator for rehabilitation robots," *Mechatronics*, vol. 23, no. 8, pp. 1072–1083, 2013.
- [5] Z. Guo, H. Yu, and L.-B. Wee, "Design of a novel compliant differential Shape Memory Alloy actuator," in *Proceedings of the 2013 26th IEEE/RSJ International Conference on Intelligent Robots and Systems: New Horizon, IROS 2013*, pp. 4925–4930, jpn, November 2013.
- [6] Z. Guo, Y. Pan, L. B. Wee, and H. Yu, "Design and control of a novel compliant differential shape memory alloy actuator," *Sensors and Actuators A: Physical*, vol. 225, pp. 71–80, 2015.
- [7] Y. Pan, Z. Guo, X. Li, and H. Yu, "Output-feedback adaptive neural control of a compliant differential SMA actuator," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2202–2210, 2017.
- [8] G. Grioli, S. Wolf, M. Garabini et al., "Variable stiffness actuators: the user's point of view," *International Journal of Robotics Research*, vol. 34, no. 6, pp. 727–743, 2015.
- [9] H. Q. Vu, X. Yu, F. Iida, and R. Pfeifer, "Improving Energy Efficiency of Hopping Locomotion by Using a Variable Stiffness Actuator," *IEEE/ASME Transactions on Mechatronics*, vol. 21, no. 1, pp. 472–486, 2016.
- [10] R. Schiavi, G. Grioli, S. Sen, and A. Bicchi, "VSA-II: A novel prototype of variable stiffness actuator for safe and performing robots interacting with humans," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation, ICRA 2008*, pp. 2171–2176, usa, May 2008.
- [11] L. C. Visser, R. Carloni, and S. Stramigioli, "Energy-efficient variable stiffness actuators," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 865–875, 2011.
- [12] A. Jafari, N. G. Tsagarakis, B. Vanderborght, and D. G. Caldwell, "A novel actuator with adjustable stiffness (AwAS)," in *Proceedings of the 23rd IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010*, pp. 4201–4206, twn, October 2010.
- [13] A. Jafari, N. G. Tsagarakis, I. Sardellitti, and D. G. Caldwell, "A new actuator with adjustable stiffness based on a variable ratio lever mechanism," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 1, pp. 55–63, 2014.
- [14] N. G. Tsagarakis, I. Sardellitti, and D. G. Caldwell, "A new variable stiffness actuator (CompAct-VSA): design and modelling," in *Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems: Celebrating 50 Years of Robotics (IROS '11)*, pp. 378–383, IEEE, San Francisco, Calif, USA, September 2011.
- [15] J. Sun, Y. Zhang, C. Zhang, Z. Guo, and X. Xiao, "Mechanical design of a compact Serial Variable Stiffness Actuator (SVSA) based on lever mechanism," in *Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 33–38, Singapore, May 2017.
- [16] G. Tonietti, R. Schiavi, and A. Bicchi, "Design and control of a variable stiffness actuator for safe and fast physical human/robot interaction," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 526–531, esp, April 2005.
- [17] G. Palli, C. Melchiorri, and A. De Luca, "On the feedback linearization of robots with variable joint stiffness," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation, ICRA 2008*, pp. 1753–1759, usa, May 2008.
- [18] G. Buondonno and A. De Luca, "Efficient Computation of Inverse Dynamics and Feedback Linearization for VSA-Based Robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 908–915, 2016.
- [19] I. Sardellitti, G. A. Medrano-Cerda, N. Tsagarakis, A. Jafari, and D. G. Caldwell, "Gain scheduling control for a class of variable stiffness actuators based on lever mechanisms," *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 791–798, 2013.
- [20] F. Petit and A. Albu-Schäffer, "State feedback damping control for a multi DOF variable stiffness robot arm," in *Proceedings of the 2011 IEEE International Conference on Robotics and Automation, ICRA 2011*, pp. 5561–5567, chn, May 2011.
- [21] F. Petit, A. Daasch, and A. Albu-Schaeffer, "Backstepping Control of Variable Stiffness Robots," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 6, pp. 2195–2202, 2015.
- [22] A. Zhakatayev, M. Rubagotti, and H. A. Varol, "Closed-Loop Control of Variable Stiffness Actuated Robots via Nonlinear Model Predictive Control," *IEEE Access*, vol. 3, pp. 235–248, 2015.
- [23] G. Palli and C. Melchiorri, "Output-based control of robots with variable stiffness actuation," *Journal of Robotics*, vol. 2011, Article ID 735407, 15 pages, 2011.
- [24] S. Ulrich, J. Z. Sasiadek, and I. Barkana, "Nonlinear adaptive output feedback control of flexible-joint space manipulators with joint stiffness uncertainties," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 6, pp. 1961–1975, 2014.

- [25] T. Dierks, B. Brenner, and S. Jagannathan, "Neural network-based optimal control of mobile robot formations with reduced information exchange," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1407–1415, 2013.
- [26] Y. Li, S. S. Ge, Q. Zhang, and T. . Lee, "Neural networks impedance control of robots interacting with environments," *IET Control Theory & Applications*, vol. 7, no. 11, pp. 1509–1519, 2013.
- [27] X. Li and C. C. Cheah, "Adaptive neural network control of robot based on a unified objective bound," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 1032–1043, 2014.
- [28] M. Hamdy and G. EL-Ghazaly, "Adaptive neural decentralized control for strict feedback nonlinear interconnected systems via backstepping," *Neural Computing and Applications*, vol. 24, no. 2, pp. 259–269, 2014.
- [29] R. Mei and C. Yu, "Adaptive Neural Output Feedback Control for Uncertain Robot Manipulators with Input Saturation," *Complexity*, Article ID 7413642, 12 pages, 2017.
- [30] B. Xu and P. Zhang, "Minimal-learning-parameter technique based adaptive neural sliding mode control of MEMS gyroscope," *Complexity*, vol. 2017, Article ID 6019175, 8 pages, 2017.
- [31] Y. Pan, Y. Liu, B. Xu, and H. Yu, "Hybrid feedback feedforward: An efficient design of adaptive neural network control," *Neural Networks*, vol. 76, pp. 122–134, 2016.
- [32] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2125–2136, 2017.
- [33] M. Hamdy, S. Abd-Elhaleem, and M. A. Fkirin, "Time-varying delay compensation for a class of nonlinear control systems over network via H ∞ adaptive fuzzy controller," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2114–2124, 2017.
- [34] M. Boukens, A. Boukabou, and M. Chadli, "Robust adaptive neural network-based trajectory tracking control approach for nonholonomic electrically driven mobile robots," *Robotics and Autonomous Systems*, vol. 92, pp. 30–40, 2017.
- [35] G. W. Woodford, M. C. du Plessis, and C. J. Pretorius, "Concurrent controller and Simulator Neural Network development for a snake-like robot in Evolutionary Robotics," *Robotics and Autonomous Systems*, vol. 88, pp. 37–50, 2017.
- [36] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.
- [37] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.
- [38] Y. Pan and H. Yu, "Biomimetic hybrid feedback feedforward neural-network learning control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 6, pp. 1481–1487, 2017.
- [39] S. Huh, G. Tonietti, and A. Bicchi, "Neural network based robust adaptive control for a variable stiffness actuator," in *Proceedings of the 2008 Mediterranean Conference on Control and Automation, MED'08*, pp. 1028–1034, Ajaccio, France, June 2008.
- [40] H. K. Khalil, *Nonlinear Systems*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2nd edition, 2002.
- [41] J. A. Farrell and M. M. Polycarpou, *Adaptive Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches*, Wiley, Hoboken, NJ, USA, 2006.
- [42] Y. Pan and H. Yu, "Composite learning from adaptive dynamic surface control," *IEEE Transactions on Automatic Control*, vol. 61, no. 9, pp. 2603–2609, 2016.
- [43] Y. Pan, T. Sun, and H. Yu, "Peaking-free output-feedback adaptive neural control under a nonseparation principle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3097–3108, 2015.

Research Article

Semiactive Nonsmooth Control for Building Structure with Deep Learning

Qing Wang,¹ Jianhui Wang,¹ Xiaofang Huang,² and Li Zhang³

¹School of Mechanical and Electric Engineering, Guangzhou University, Guangzhou 510006, China

²Engineering Earthquake Resistance Center, Guangzhou University, Guangzhou 51045, China

³Guangzhou Real Estate Management Vocational School, Guangzhou 510320, China

Correspondence should be addressed to Jianhui Wang; 87517619@qq.com

Received 13 July 2017; Revised 2 October 2017; Accepted 12 October 2017; Published 6 November 2017

Academic Editor: Guang Li

Copyright © 2017 Qing Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Aiming at suppressing harmful effect for building structure by surface motion, semiactive nonsmooth control algorithm with Deep Learning is proposed. By finite-time stable theory, the building structure closed-loop system's stability is discussed under the proposed control algorithm. It is found that the building structure closed-loop system is stable. Then the proposed control algorithm is applied on controlling the building structural vibration. The seismic action is chosen as El Centro seismic wave. Dynamic characteristics have comparative analysis between semiactive nonsmooth control and passive control in two simulation examples. They demonstrate that the designed control algorithm has great robustness and anti-interference. The proposed control algorithm is more effective than passive control in suppressing structural vibration.

1. Introduction

Seismic isolation technology for building structure has made great progress. And seismic isolation technology is used widely in many countries. However, seismic isolation system is along with large displacement when this system reduces seismic response of superstructure effectively. Thus, the application of seismic isolation technology is limited. In addition, the earthquake has strong randomness. If the designed passive isolation is used to suppress vibration, isolation parameters, which are used in earthquake action of some characteristics, not always have satisfying damping effect for other different characteristics earthquake. Hence, many scholars use control strategies to reduce displacement of isolating system. At the same time, the structural response is not added [1].

The study in regard to structural vibration control algorithm is a hot topic of structural vibration control field all the time [2]. In the last thirty years, structural vibration control technologies have made great progress. The technologies gradually become available means which ensure security of structure and resist the natural disasters. The so-called

structural vibration control is used to change and adjust structural dynamic characteristic by deploying or inlaying some device or some equipment that applies force. And structural vibration control can obstruct and offset external excitation or dissipate energies that input to the structural system. The control actions can achieve the aim of controlling the structural dynamic characteristic reasonably. According to the characteristics of earthquake action, the structure that applied control measures possesses self-regulating ability of resisting earthquake action. Thereby, the structural intelligence comes true.

On the basis of having extra powers or not, the structural vibration control is divided into three classes [3]. The first class is passive control. The passive control does not need extra power. The structural vibration is suppressed only by interaction of control device and structure. Then the second class is active control that needs high-power extra power to drive actuator. The control is decided by measured excitation and feedback of structural response. The third class is semiactive control, which does not need to input great powers. Just a little power is applied when structural response is up to boundary value. And control device is switched to

working state to reach parameters that adjust control system and the aim of reducing structural response.

The premise and foundation of structural vibration control are control algorithm. Therefore, according to features external disturbance, an important characteristic of structural vibration control is giving real control force by preset control algorithm. It guarantees that structural response is controlled to the safe range. And optimal control algorithm [4] gains lots of applications in civil structural control. However, the optimal control problem of forced vibration that earthquakes cause in long time does not establish extremum condition of optimal control. Thus, with the aid of COC (Classical Linear Optimal Control) [5], IOC (Instantaneous Optimal Control) [6], and LQG (Linear Quadratic Gaussian) control, the problem is improved. The other representative control algorithms [7–12] are LQR (Linear Quadratic Regulator) control, ILC (Iterative Learning Control), fuzzy logic control, and so on. Nevertheless, the aforementioned results still have large parameter perturbation of system and are over-sensitive to external interference. It is extremely important to develop and apply the structural vibration control algorithm with stronger robustness. This kind of control algorithm has a little influence on the system parameters and has less sensitivity to external disturbances. After the 1990s, the American scholar Yang and others took the lead in the application of sliding mode control to vibration control of civil engineering structure. Sliding mode control, a switching control, makes the closed-loop system have the characteristics of finite-time convergence and strong robustness. But it is discontinuous for state variable of system. High-speed switching of control causes chattering phenomenon in the system. Nonsmooth control is able to achieve finite-time stability of system state. The robustness and inter-inference performance of system are both enhanced. Compared with traditional sliding mode control, nonsmooth control is continuous with respect to state variable. The control method receives attention and gains application [13–15] widely due to distinct advantages. Now nonsmooth control is studied and applied in the areas of design of high-precision guidance laws [16], attitude control of spacecraft [17], control of permanent magnet synchronous motor position [18], and so on.

However, nonsmooth control is not able to express uncertain information of system. Deep Learning is deep neural networks of a class of special nonclassified tags. Deep Learning usually has stackable denoising autoencoders and transformational autoencoders. The dimensions of their output vector and input vector are the same in some form of input vector generally. Expression of a data is studied or initial data is coded effectively by hidden layer. According to the study of literatures [19–29], Deep Learning can gain more representative characteristic information by training large scale data. Thus, the sample may be classified and estimated to improve precision of information.

In this paper, the nonlinear model of the building structure is described. According to expression of uncertain information with Deep Learning, the stability is deduced and analyzed in the based isolation structure by applying semiactive nonsmooth control algorithm. Under seismic wave called El Centro, the numerical simulation of two

different examples is simulated numerically with semiactive nonsmooth control and the passive control. According to the comparison of simulation results, the feasibility and validity of the algorithm are verified.

2. Mathematical Models and Related Theories

2.1. Passive Isolation Structure. Some researchers, such as Zhou [4], Tang and Lium [30], and Kelly [31], do lots of work in seismic responses of passive isolation structure. It sets flexible seismic isolation between the bottom of superstructure and the top of foundation. Seismic isolation is used to prevent the superstructure from seismic energy [4]. The seismic isolation not only reduces seismic dynamic response of superstructure, but also ensures security of major structure and in-house facility effectively. In addition, the isolation system increases the natural vibration period. Due to increasing the natural period of the structure that exceeds the excitatory period of earthquake [4], isolation system avoids the phenomena of resonance and near resonance. Thus, isolation system reduces the seismic acceleration response of the structure.

The key to dynamic analysis of a building structure is the simplification of the structure. Two methods of structural simplification are described [32]. One is the way of lumped-mass; another one is the way of the centralized quality. If the distribution quality method is used to analyze dynamic, the structural equation of motion is a partial differential equation. It is not conducive for research of the practical application. Thus, the structural quality is often described by the lumped-mass method in practical engineering. It is simplified as a multi-degree-of-freedom story shear model. The superstructure and foundation of building are separated by seismic isolation. The energy, which is generated by seismic waves during the earthquake, is transferred to the seismic isolation from the base. And the seismic isolation prevents superstructure by self-deformation and vibration to dissipate energy [33].

In general, the horizontal stiffness of the seismic isolation bearing is much smaller than its vertical stiffness. While dynamic responses are analyzed, seismic isolation bearing is approximated to horizontal movement. And the vertical deformation that is caused by the swing is ignored. Compared with traditional structure, dynamic analysis of isolation structure consists of two parts. It consists of the isolation layer and the superstructure. In this paper, the building structure is simplified as one-dimensional story shear model. Under the action of horizontal earthquake, the equation of motion is represented by the following differential equation:

$$M\ddot{X} + C\dot{X} + KX = -MR\ddot{x}_g. \quad (1)$$

In this equation, $X = [x_b \ x_1 \ x_2 \ \cdots \ x_i \ \cdots \ x_n]^T$ is displacement vector of the structure, where x_b is base displacement relative to the ground and x_i is displacement of the i th floor relative to the ground. \dot{X} and \ddot{X} represent velocity vector and acceleration vector that include base velocity and acceleration. \ddot{x}_g is a vector of ground acceleration. R is a matrix of influence coefficient. M , K , and C are mass matrix,

proportional damping matrix, and stiffness matrix of the structure that include parameters of the isolation layer.

$$\begin{aligned}
 M &= \begin{bmatrix} m_b & & & & \\ & m_1 & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & m_n \end{bmatrix} \\
 K &= \begin{bmatrix} k_b + k_1 & -k_1 & & & \\ k_1 & k_1 + k_2 & -k_2 & & \\ & & \ddots & \ddots & \\ & & & -k_{n-1} & k_{n-1} + k_n & -k_n \\ & & & & -k_n & k_n \end{bmatrix} \\
 C &= \begin{bmatrix} c_b + c_1 & -c_1 & & & \\ c_1 & c_1 + c_2 & -c_2 & & \\ & & \ddots & \ddots & \\ & & & -c_{n-1} & c_{n-1} + c_n & -c_n \\ & & & & -c_n & c_n \end{bmatrix}.
 \end{aligned} \tag{2}$$

2.2. Finite-Time Stability [14]. Semiactive nonsmooth control, which is introduced for the control system, can effectively improve the performance of the system. It realizes the finite-time stability of the system state and improves its robustness and anti-interference of the system. Combined with self-learning of the neural networks [34–37], note of hidden players is considered and the whole intelligence degree of the system is enhanced.

The finite-time stability problem of the following nonlinear systems is considered as

$$\dot{z} = f(z), \quad f(0) = 0, \quad z \in R^n. \tag{3}$$

In this equation, a vector field of C^0 is $f(z)$.

Theorem 1. *The homogeneous dilations of $z \in R^n$ are as follows:*

$$\Delta_\varepsilon^\rho(z) = [\varepsilon^{\rho_1} z_1, \varepsilon^{\rho_2} z_2, \dots, \varepsilon^{\rho_n} z_n]^T, \tag{4}$$

where $\forall \varepsilon > 0, \rho_i > 0 \ i = 1, 2, \dots, n$.

Theorem 2. *For a vector field $f(z)$ in system (3), suppose there exists $l \in R$, and the function is satisfied as*

$$f_i(\Delta_\varepsilon^\rho(z)) = \varepsilon^{l+\rho_i} f_i(z) \quad i = 1, 2, \dots, n. \tag{5}$$

Then the vector field $f(z)$ is called vector fields homogeneous and related system is called homogeneous system. The homogeneous degree for the system is l .

Theorem 3. *If system (3) is asymptotically stable and has convergence characteristic of finite-time, which means that the system state converges to equilibrium point of the system in finite-time, the system is finite-time stable. Equilibrium point 0*

of system (3) has locally asymptotic stability. Equilibrium point 0 that is assumed is asymptotically global stability, while system (3) is a homogeneous system. And if the homogeneous degree in system is half negative ($l < 0$), system (3) is finite-time stable.

3. Design of Semiactive Nonsmooth Control Algorithm with Deep Learning

Passive control system can effectively reduce the loss caused by earthquakes, whereas the random of large earthquake is controlled difficultly. If seismic isolation produces the large displacement, its effect may be lost, which affects the whole performance of whole structure. In order to control the large displacement in the isolation layer, nonsmooth control algorithm is studied and designed. It can restrain the seismic wave that causes influence for building structural stability. In order to study the control algorithm conveniently, the maximum permissible displacement is assumed to be at the zero point. The related analysis is conducted.

3.1. Design of Nonsmooth Algorithm with Deep Learning. With the action of one-dimensional horizontal seismic acceleration, the equation of motion is as

$$M\ddot{X} + C\dot{X} + KX = -MR\ddot{x}_g + EU. \tag{6}$$

In this equation, E is a position matrix of controllers. U is a vector comprising control forces. The acceleration \ddot{x}_g is unknown disturbance and established by Deep Learning neural network. In this paper, autoencoder neural network (see [20, 21]) is used to establish \ddot{x}_g . Other parameters have been explained in (1).

Defining a state-space vector $Y = (Y_1 \ Y_2)^T$, where $Y_1 = (Y_{11} \ Y_{12} \ \dots \ Y_{1n})^T = X$, $Y_2 = (Y_{21} \ Y_{22} \ \dots \ Y_{2n})^T = \dot{X}$. The equation of motion is rewritten as [6, 38]

$$\dot{Y} = AY + W\ddot{x}_g + BU \tag{7}$$

$$A = \begin{bmatrix} 0 & I_{n \times n} \\ -M^{-1} \cdot K & -M^{-1} \cdot C \end{bmatrix}$$

$$W = \begin{bmatrix} 0 \\ M^{-1} \cdot F \end{bmatrix} \tag{8}$$

$$B = \begin{bmatrix} 0 \\ M^{-1} \cdot E \end{bmatrix},$$

where A , W , and E are state matrices. I is a unit matrix. According to the rank criterion, the system (see (7)) is observable and controllable. Therefore, structural vibration can be effectively suppressed by designing U properly.

It is considered that the actuator would be only installed at isolation layer. For state equation (6), the maximum displacement value of the isolation layer, which is assumed, is s . Using $\underline{v} = x - \text{sgn}(x) \cdot s$, obviously,

$$\ddot{x} = \ddot{\underline{v}}. \tag{9}$$

From the above, the following can be achieved:

$$(m\ddot{v} + c\dot{v} + kv) = -m\ddot{x}_g + u. \quad (10)$$

Set $z = (z_1, z_2)^T = (v, \dot{v})^T$, and choose $u = c\dot{v} + kv + m\ddot{x}_g + mv_1$; the following is achieved:

$$\dot{z} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \cdot z + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot v_1. \quad (11)$$

The system is rewritten as follows:

$$\begin{aligned} \dot{z}_1 &= z_2 \\ \dot{z}_2 &= v_1. \end{aligned} \quad (12)$$

Let

$$v_1 = -k_1 \cdot \text{sgn}(z_1) \cdot |z_1|^{\alpha_1} - k_2 \cdot \text{sgn}(z_2) \cdot |z_2|^{\alpha_2}, \quad (13)$$

where $k_1 > 0$, $k_2 > 0$, $0 < \alpha_1 < 1$, $\alpha_2 = 2\alpha_1/(1 + \alpha_1)$, $i = 1, 2, \dots, n$.

3.2. Stable Analysis of Nonsmooth Algorithms. Lyapunov function of the subsystem is selected as

$$F_1(z_1, z_2) = \frac{k_1}{1 + \alpha_1} |z_1|^{1 + \alpha_1} + \frac{1}{2} z_2^2. \quad (14)$$

The derivation of (14) is as follows:

$$\dot{F}_1(z_1, z_2) = -k_2 |z_2|^{1 + \alpha_2}. \quad (15)$$

Obviously, $\dot{F}_1(z_1, z_2)$ is half negative. Therefore, the system is stable. The following set is considered as

$$V_1 = \{(z_1, z_2) : \dot{F}_1(z_1, z_2)\}. \quad (16)$$

By (11), the following is achieved:

$$V_1 = \{(z_1, z_2) : z_2 = 0\}. \quad (17)$$

Let $z_2(t) = 0$; then we have $\dot{z}_1(t) = \dot{z}_2(t) = 0$. When we substitute it into the closed-loop equations of system, it could be achieved; it is $z_1(t) = 0$. Hence, the contained largest invariant set of the set V_1 is as follows:

$$(z_1 \ z_2) = (0 \ 0). \quad (18)$$

According to the invariance principle, in the equilibrium point $(z_1 \ z_2) = (0 \ 0)$, the system is globally asymptotically stable and $f(z_1, z_2) = [f_1, f_2]^T = [\dot{z}_1, \dot{z}_2]^T$ is set. According to the theorem of homogeneous system [13], if the homogeneity of f is satisfied, the formula is established as follows:

$$\begin{aligned} & f(\varepsilon^{\rho_1} z_1, \varepsilon^{\rho_2} z_2) \\ &= \begin{bmatrix} \varepsilon^{\rho_2} y_{2i} \\ -k_1 \varepsilon^{\rho_1 \alpha_1} \text{sgn}(z_1) |z_1|^{\alpha_1} - k_2 \varepsilon^{\rho_2 \alpha_2} \text{sgn}(z_2) |z_2|^{\alpha_2} \end{bmatrix} \\ &= \begin{bmatrix} \varepsilon^{\rho_1 + l} & 0 \\ 0 & \varepsilon^{\rho_2 + l} \end{bmatrix} \begin{bmatrix} y_{2i} \\ -k_1 \text{sgn}(z_1) |z_1|^{\alpha_1} - k_2 \text{sgn}(z_2) |z_2|^{\alpha_2} \end{bmatrix} \quad (19) \\ &= \begin{bmatrix} \varepsilon^{\rho_1 + l} f_1(z_1, z_2) \\ \varepsilon^{\rho_2 + l} f_2(z_1, z_2) \end{bmatrix}. \end{aligned}$$

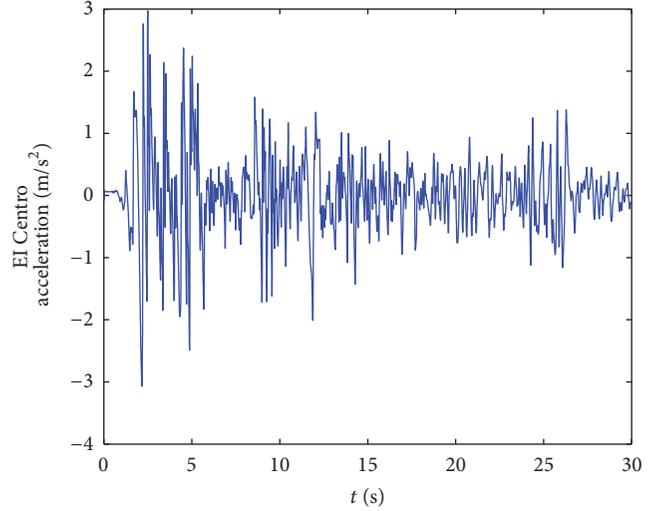


FIGURE 1: Seismic wave.

It can be got as

$$\begin{aligned} \rho_1 + l &= \rho_2 \\ \rho_2 + l &= \rho_1 \cdot \alpha_1 = \rho_2 \cdot \alpha_2. \end{aligned} \quad (20)$$

Let $\rho_1 = 1$, $\rho_2 = (1 + \alpha_1)/2$. Equation (6) is established and $f(z_1, z_2)$ is satisfied with homogeneity. It means that the system is a homogeneous system. The degree of homogeneity in the system is $l = (\alpha_1 - 1)/2 < 0$ which means negative degree of the homogeneity. Thus, system (9) is globally finite-time stable.

4. Numerical Results of Examples

In order to illustrate the effectiveness of the semiactive nonsmooth control algorithm, two three-layer base isolation structure models are analyzed with different parameters.

Model parameters of example 1 are as follows.

The masses of superstructure are $m_1 = m_2 = m_3 = 9.167 \times 10^5$ kg. The stiffness of superstructure is $k_1 = k_2 = k_3 = 1.54 \times 10^6$ kN/m. The damping coefficients of the superstructure are $c_1 = c_2 = c_3 = 4.813 \times 10^7$ kNs/m. The mass of the isolation layer is $m_b = 8.74 \times 10^5$ kg. The stiffness of the isolation layer is $k_b = 0.056 \times 10^9$ kN/m. The damping coefficient of the isolation layer is $c_b = 0.832 \times 10^7$ kNs/m. The requirement of horizontal displacement is 20 mm.

El Centro Seismic wave, the unknown disturbance, is estimated by deep neural network. The result is shown in Figure 1 and peak value of acceleration is 3 m/s^2 . According to the parameters of the aforementioned model, the dynamic responses are simulated in two ways of semiactive nonsmooth control and passive control. The displacements, velocities, and acceleration and control forces of the seismic isolation are shown in Figures 2–4. In order to further analyze the effect of semiactive nonsmooth control and passive control, the maximum displacement $x_{b\max}$ and the maximum acceleration $a_{b\max}$ in the seismic isolation are made statistics. The results are shown in Table 1.

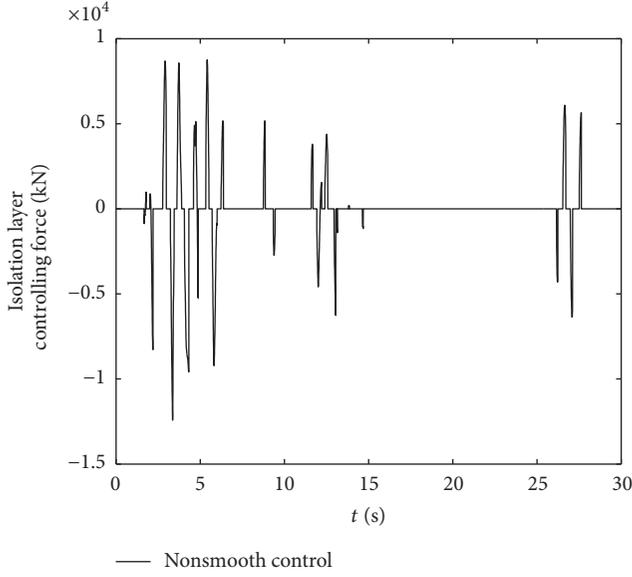


FIGURE 2: Control force response of isolation layer.

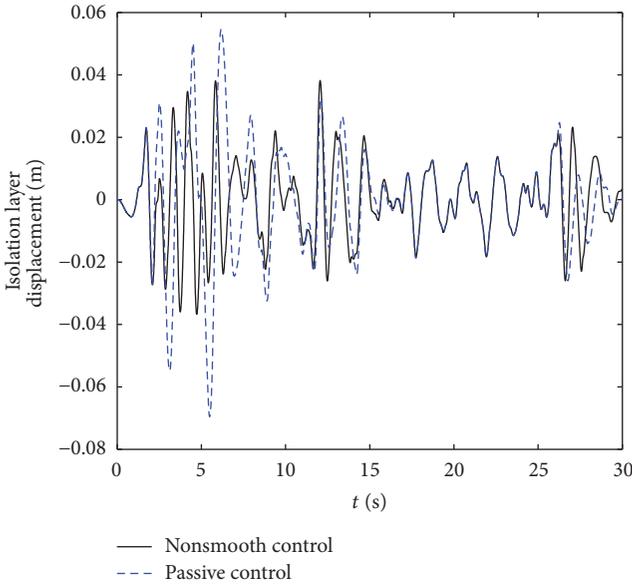


FIGURE 3: Displacement response of isolation layer.

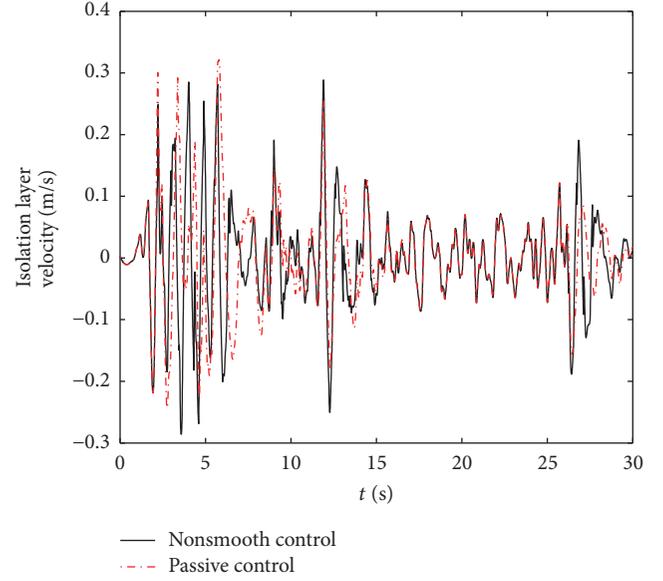


FIGURE 4: Velocity response of isolation layer.

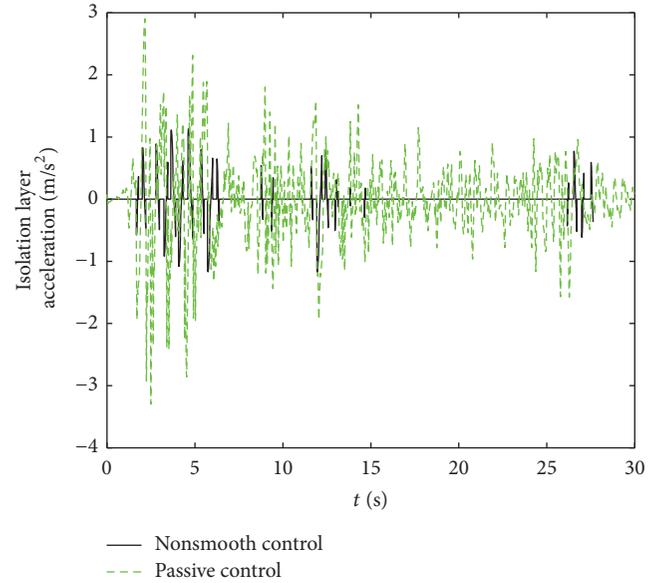


FIGURE 5: Acceleration response of isolation layer.

TABLE 1: Comparison of peak data.

Control strategy	$x_{b\max}$ (m)	$a_{b\max}$ (m/s^2)
Isolation layer		
Passive control	0.070	3.300
Nonsmooth semiactive control	0.038	1.171

As shown in Figures 1–5, in the range of the external control force, the velocity of isolation layer with semiactive nonsmooth control is a little larger than passive control. But in the terms of cutting down the displacements and acceleration of isolation layer, semiactive nonsmooth control is superior to passive control. As is shown in Table 1, the peak

acceleration of isolation layer with semiactive nonsmooth control is down to 35%. Compared with passive control, the peak displacement of isolation layer is decreased by 46%. In the isolation layer, the horizontal distortion is reduced drastically. Hence, control effect is obvious.

Model parameters of example 2 are as follows.

The masses of superstructure are $m_1 = 53200$ kg, $m_2 = 61300$ kg, and $m_3 = 54500$ kg. The stiffness of superstructure is $k_1 = 115.3 \times 10^3$ kN/m, $k_2 = 115.3 \times 10^3$ kN/m, and $k_3 = 74.9 \times 10^3$ kN/m. The damping coefficients of superstructure are $c_1 = 171.5$ kNs/m, $c_2 = 171.5$ kNs/m, and $c_3 = 111.4$ kNs/m. The mass of isolation layer is $m_b = 41 \times 10^3$ kg. The stiffness of isolation layer is $k_b = 1.5 \times 10^3$ kN/m. The

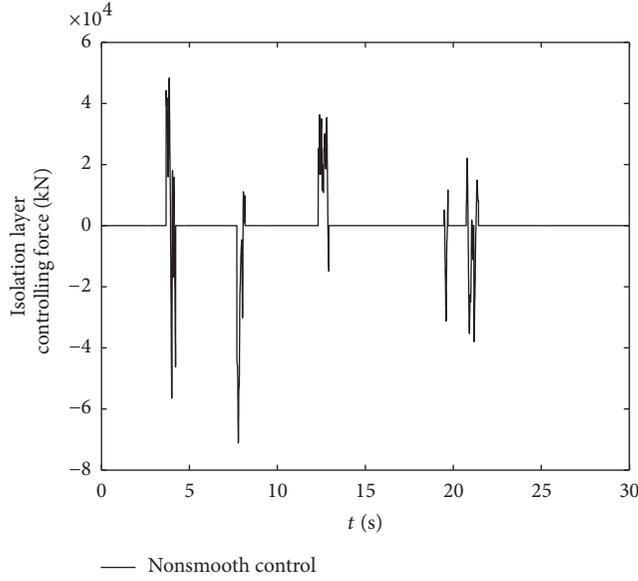


FIGURE 6: Control force response of isolation layer.

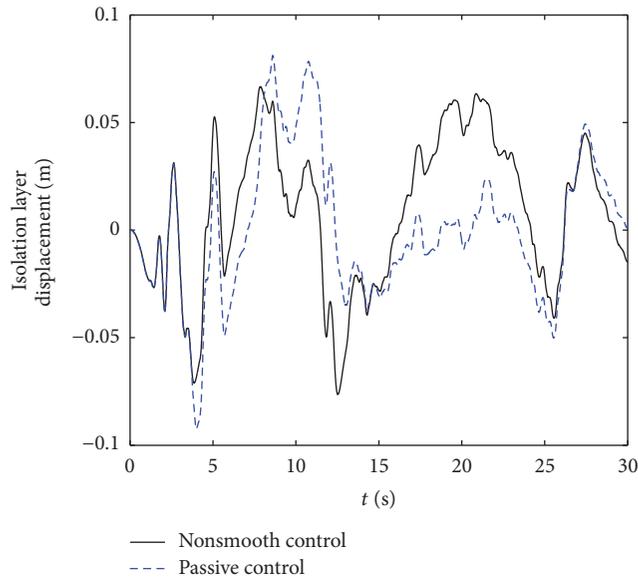


FIGURE 7: Displacement response of isolation layer.

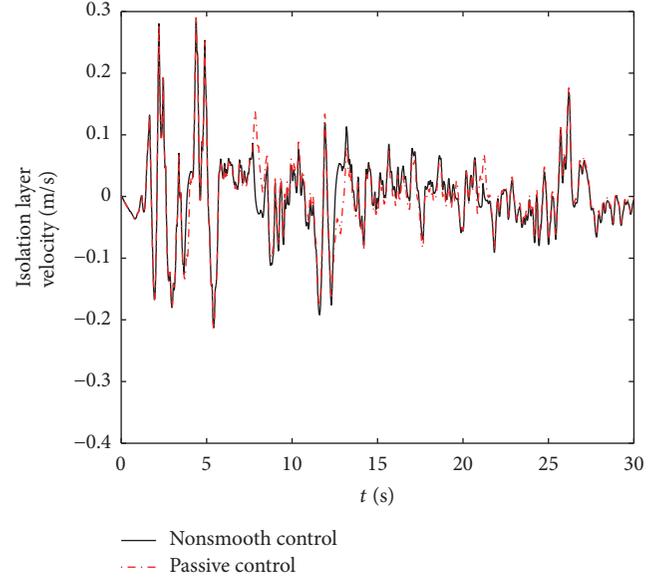


FIGURE 8: Velocity response of isolation layer.

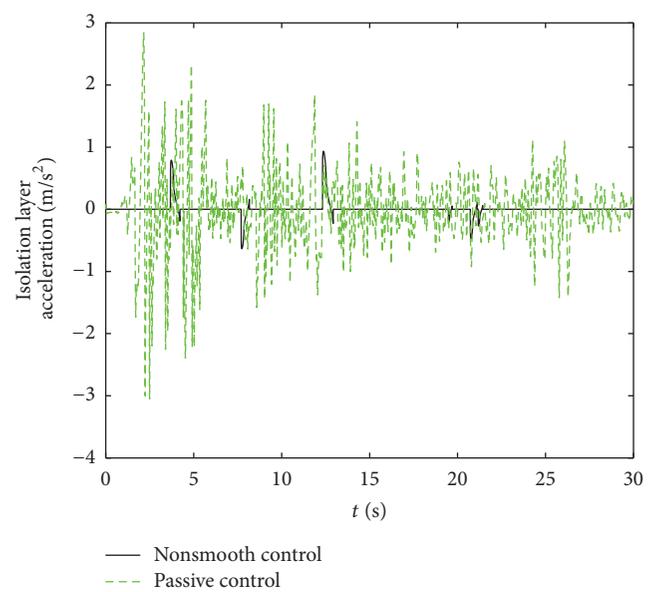


FIGURE 9: Acceleration response of isolation layer.

damping coefficient of the isolation layer is $c_b = 53.5$ kNs/m. The requirement of horizontal displacement is 60 mm.

Example 2 inputs seismic wave El Centro and is simulated and analyzed. The requirement of horizontal displacement is a criterion. And the proposed model is in the two conditions, respectively, semiactive nonsmooth control and passive control. The contrastive curves, which include displacements, velocities, and acceleration, are shown in Figures 7–9. The peak displacement $x_{b\max}$ and the peak acceleration $a_{b\max}$ in the isolation layer of the proposed model are counted. They are showed in Table 2.

As is shown in time-history curves in Figures 6–9, the conclusion can be drawn similarly. The displacement and peak displacement in the isolation layer are obviously

TABLE 2: Comparison of peak data.

Control strategy	$x_{b\max}$ (m)	$a_{b\max}$ (m/s ²)
Isolation layer		
Passive control	0.0925	3.0703
Nonsmooth semiactive control	0.0764	0.9333

decreased in the semiactive nonsmooth control system. As is shown in Table 2, the acceleration of the model in isolation layer is also declined. It means the peak acceleration is controlled. Compared with the passive control system, the peak displacement of the isolation layer decreases by 21% with semiactive nonsmooth control system. The horizontal

peak displacement in isolation layer is declined drastically. It shows that semiactive nonsmooth control algorithm is able to improve performance of passive isolation structure to prevent isolation layer from too much deformation.

From the simulated analyses of two models with different parameters, the semiactive nonsmooth control algorithm can cut down dynamic response more effective than passive control. The displacement of the structure is controlled to allowable range as much as possible. The disadvantage of excessive displacement in the isolation layer is solved effectively, so the effect of designed control is remarkable.

5. Conclusions

Semiactive nonsmooth control with Deep Learning is more effective than passive control. By using deep neural network, the designed control method not only considers the unknown term but also cuts down the structural peak response that is caused by earthquake. The analysis of seismic response in the controlled structure shows that the proposed algorithm improves the performance of isolated structure and reduces the seismic response in the isolation layer and superstructure. The designed control method prevents building structure from oversize deformation that causes structural failure. Thus, the designed control method is high-effect. Semiactive nonsmooth control algorithm has great robustness and can restrain the influence of seismic wave effectively. Thus, the designed nonsmooth control algorithm with Deep Learning is more intellectualized, feasible, and effective. The proposed control method could be theoretical basis for practical building structural engineering application.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work is supported by National Natural Science Foundation (NNSF) of China (under Grant no. 51478132) and Guangzhou City College Scientific Research Project (under Grant no. 120163017).

References

- [1] S. Etedali, M. R. Sohrabi, and S. Tavakoli, "Optimal PD/PID control of smart base isolated buildings equipped with piezoelectric friction dampers," *Earthquake Engineering and Engineering Vibration*, vol. 12, no. 1, pp. 39–54, 2013.
- [2] J. Yao, "Concept of structural control," *Journal of the Structural Division*, vol. 98, no. 7, pp. 1567–1574, 1972.
- [3] H. Hu, D. Guo, and J. Weng, "Recent advances in semi-active control of vibration," *Journal of Vibration, Measurement & Diagnosis*, vol. 4, pp. 235–244, 2001.
- [4] F. Zhou, *Engineering Structure Vibration Control*, Seismological Press, Beijing, China, 1997.
- [5] X.-M. Li and Y.-F. Du, "Curved girder bridges' control based on sequential optimal control algorithm under two-directional horizontal earthquake," *Journal of Vibration & Shock*, vol. 34, no. 10, pp. 6–33, 2015.
- [6] K. K. F. Wong and R. Yang, "Predictive instantaneous optimal control of inelastic structures during earthquakes," *Earthquake Engineering & Structural Dynamics*, vol. 32, no. 14, pp. 2179–2195, 2010.
- [7] J. Ou, *Structural Vibration Control-Active, Semi-Active and Intelligent Control*, Science Press, Beijing, China, 2003.
- [8] S. Tong and H. Tang, "Iterative learning instantaneous optimal control of discrete systems optimization of actuator positions," *Applied Mathematics and Mechanics*, vol. 37, no. 2, pp. 160–172, 2016.
- [9] M. Battaini, F. Casciati, and L. Faravelli, "Fuzzy control of structural vibration. An active mass system driven by a fuzzy controller," *Earthquake Engineering & Structural Dynamics*, vol. 27, no. 11, pp. 1267–1276, 1998.
- [10] J. N. Yang, J. C. Wu, K. Kawashima, and S. Unjoh, "Hybrid control of seismic-excited bridge structures," *Earthquake Engineering & Structural Dynamics*, vol. 24, no. 11, pp. 1437–1451, 1995.
- [11] J. H. Wang, Z. Liu, C. Chen, and Y. Zhang, "Fuzzy adaptive compensation control of uncertain stochastic nonlinear systems with actuator failures and input hysteresis," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–12, 2017.
- [12] X. Lin and R. Zhang, " H_{∞} control for stochastic systems with Poisson jumps," *Journal of Systems Science and Complexity*, vol. 24, no. 4, pp. 683–700, 2011.
- [13] K.-M. Ma, "Design of continuous non-smooth attitude control laws for spacecraft," *Journal of Astronautics*, vol. 33, no. 6, pp. 713–719, 2012.
- [14] Q. Wang, J. Wang, K. Ma et al., "Nonsmooth control algorithm for structural vibration control and its applications," *Earth Engineering and Engineering Dynamics*, vol. 36, no. 5, pp. 207–214, 2016.
- [15] J. Wang, Q. Wang, and K. Ma, "Non-smooth controller design for permanent magnet synchronous motors," *Computer Simulation*, vol. 33, no. 3, pp. 227–230, 2016.
- [16] K.-M. Ma, "Non-smooth design and implementation of high-precision guidance laws," *Journal of Ballistics*, vol. 25, no. 2, pp. 1–5, 2013.
- [17] K. Ma, "Design of continuous non-smooth attitude control laws for spacecraft," *Journal of Ballistics*, vol. 33, no. 6, pp. 713–719, 2016.
- [18] H. Liu, S. Ding, S. Li et al., "Finite-time control of PSPM position servo system," *Electric Machines and Control*, vol. 13, no. 3, pp. 424–430, 2009.
- [19] Y. Bengio, É. ThibodeauLaufer, G. Alain et al., "Deep generative stochastic networks trainable by backprop," *Computer Science*, vol. 2, pp. 226–234, 2014.
- [20] L. Deng, M. Seltzer, D. Yu et al., "Binary coding of speech spectrograms using a deep auto-encoder," in *Proceedings of the Conference of the International Speech Communication Association (INTERSPEECH '10)*, pp. 1692–1695, Makuhari, Chiba, Japan, September 2010.
- [21] G. E. Hinton, A. Krizhevsky, and S. D. Wang, "Transforming auto-encoders," in *Artificial Neural Networks and Machine Learning*, vol. 6791 of *Lecture Notes in Computer Science*, pp. 44–51, 2011.
- [22] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Computation*, vol. 23, no. 7, pp. 1661–1674, 2011.

- [23] O. Vinyals, Y. Jia, L. Deng et al., "Learning with recursive perceptual representations," *Advances in Neural Information Processing Systems*, pp. 2834–2842, 2012.
- [24] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–12, 2017.
- [25] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [26] H. Xiao, Z. Li, C. Yang et al., "Robust stabilization of a wheeled mobile robot using model predictive control based on neurodynamics optimization," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 505–516, 2017.
- [27] C. Yang, X. Wang, Z. Li et al., "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems Man & Cybernetics Systems*, vol. PP, no. 99, pp. 1–12, 2017.
- [28] C. Yang, J. Luo, Y. Pan et al., "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems Man & Cybernetics Systems*, vol. PP, no. 99, pp. 1–12, 2017.
- [29] X. Z. Meng, S. N. Zhao, T. Feng, and T. H. Zhang, "Dynamics of a novel nonlinear stochastic SIS epidemic model with double epidemic hypothesis," *Journal of Mathematical Analysis and Applications*, vol. 433, no. 1, pp. 227–242, 2016.
- [30] J. Tang and Z. Lium, *Isolated Structure Design*, Huazhong University of Science and Technology Press, Wuhan, China, 1993.
- [31] J. M. Kelly, *Earthquake-Resistant Design with Rubber*, Springer, London, England, 1993.
- [32] G. Li, J. Li, and X. Su, *Seismic Design of Building Structures*, China Building Industry Press, 2nd edition, 2007.
- [33] M. Liu, X. Tian, D. Wang et al., "Research and application of building structure base isolation technology," *Technology for Earthquake Disaster Prevention*, vol. 1, no. 1, pp. 31–38, 2006.
- [34] Z. Zhao, X. Wang, C. Zhang, Z. Liu, and J. Yang, "Neural network based boundary control of a vibrating string system with input deadzone," *Neurocomputing*, In Press.
- [35] F. Wang, B. Chen, C. Lin, J. Zhang, and X. Meng, "Adaptive neural network finite-time output feedback control of quantized nonlinear systems," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–10, 2017.
- [36] H. Cheng and T. Zhang, "A new predator-prey model with a profitless delay of digestion and impulsive perturbation on the prey," *Applied Mathematics and Computation*, vol. 217, no. 22, pp. 9198–9208, 2011.
- [37] X. Dong, Z. Bai, and S. Zhang, "Positive solutions to boundary value problems of p -Laplacian with fractional derivative," *Boundary Value Problems*, vol. 2017, article 5, 2017.
- [38] Z. Bai, S. Zhang, S. Sun, and C. Yin, "Monotone iterative method for fractional differential equations," *Electronic Journal of Differential Equations*, vol. PP, no. 99, pp. 1–12, 2016.

Research Article

Neural Learning Control of Flexible Joint Manipulator with Predefined Tracking Performance and Application to Baxter Robot

Min Wang, Huiping Ye, and Zhiguang Chen

School of Automation Science and Engineering, Guangzhou Key Laboratory of Brain Computer Interaction and Applications, South China University of Technology, Guangzhou 510641, China

Correspondence should be addressed to Min Wang; auwangmin@scut.edu.cn

Received 20 July 2017; Accepted 14 September 2017; Published 31 October 2017

Academic Editor: Yanan Li

Copyright © 2017 Min Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper focuses on neural learning from adaptive neural control (ANC) for a class of flexible joint manipulator under the output tracking constraint. To facilitate the design, a new transformed function is introduced to convert the constrained tracking error into unconstrained error variable. Then, a novel adaptive neural dynamic surface control scheme is proposed by combining the neural universal approximation. The proposed control scheme not only decreases the dimension of neural inputs but also reduces the number of neural approximators. Moreover, it can be verified that all the closed-loop signals are uniformly ultimately bounded and the constrained tracking error converges to a small neighborhood around zero in a finite time. Particularly, the reduction of the number of neural input variables simplifies the verification of persistent excitation (PE) condition for neural networks (NNs). Subsequently, the proposed ANC scheme is verified recursively to be capable of acquiring and storing knowledge of unknown system dynamics in constant neural weights. By reusing the stored knowledge, a neural learning controller is developed for better control performance. Simulation results on a single-link flexible joint manipulator and experiment results on Baxter robot are given to illustrate the effectiveness of the proposed scheme.

1. Introduction

Due to the great demands in industrial applications, the tracking control problem for flexible joint robot (FJR) manipulator has attracted much attention in recent years. Unlike rigid joint robot, the joint flexibility of FJR results in complex control situation, so that the control problem of FJR becomes much more difficult. In the past few decades, lots of efforts have been made on the research of FJR systems. Based on the model of FJR presented in [1], multifarious nonlinear control methods are presented such as backstepping method [2–4], sliding-mode control [5–8], switching control [9], fuzzy control [10], and neural network control [11, 12]. In consideration of the problem caused by the inherent structure of FJR under practical circumstance, such as friction, time delay, and variable stiffness, some researchers proposed effective strategies to solve such problem [13–15]. Moreover, the teleoperation control method is also widely used in robot research [16, 17].

The backstepping control [18] is known as one of the popular method for designing the control scheme of FJR. Nevertheless, it should be pointed out that this method has a drawback called “explosion of complexity” [19]. This problem generally occurs in the design of neural networks (NNs) during backstepping procedure. To overcome this problem, some researchers used the intermediate variables as neural inputs to reduce the dimension of neural network input vector [20]. The method in [20] did work well but the problem remained unsolved. Then other researchers proposed a dynamic surface control (DSC) method by introducing a first-order filter at each step of the backstepping procedure [21]. Due to the property of DSC method, many researchers presented their control schemes combined with DSC method [22–27]. In [24], a new robust output feedback control approach for flexible joint electrically driven robots via the observer-based dynamic surface method was proposed, which only requires position measurement of the system.

Besides, the transient and steady-state tracking performance constraints of system's output are an important issue that needs to be taken into consideration [28, 29]. According to the practical operating environment, the manipulator is not only demanded to trace the reference trajectory accurately but also required to keep the tracking error within a specified range. To satisfy this condition, a performance function transformation was used to convert the "constrained" system into the "unconstrained" one [30]. Based on the idea in [30], further researches on prescribed performance for a variety of systems are proposed [31–36]. Authors in [31, 32] presented novel controllers for FJRs to achieve tracking control of link angles with any prescribed performance requirements. By combining neural learning control scheme, further results are given in [33–35]. In [36], an adaptive prescribed performance tracking control scheme is investigated for a class of output feedback nonlinear systems with input unmodeled dynamics based on dynamic surface control method.

In addition, adaptive neural control of nonlinear system has been widely studied for decades, but most of the traditional works focus on the system stability through online adjustment of neural weights and less works discuss the knowledge acquisition, storage, and utilization of optimal neural weights. To achieve such learning ability, the key problem is to verify the persistent excitation (PE) condition. Thanks to the results in [37], a deterministic learning mechanism is proposed, which proved the satisfaction of PE condition for the localized radial basis function (RBF) NN centered in a neighborhood along recurrent orbits. The result was extended to nonlinear systems satisfying matching conditions [38–40]. By combining recursive design technologies such as backstepping control and the system decomposition strategy, the deterministic learning was also applied to solve learning problem of accurate identification of ocean surface ship and robot manipulation in uncertain dynamical environments [41–44]. However, due to the recursive property of backstepping control, the convergence of neural weights has to be recursively verified based on the system decomposition strategy. It would be a tedious and complex process since the intermediate variables grow drastically as the order of system increases. Therefore, it is difficult to prove all neural weights convergence for high-order system by existing works.

This paper focuses on learning from adaptive neural control of flexible joint manipulator with unknown dynamics under the prescribed constraints. A performance function is introduced to transform the constrained tracking error into the unconstrained variable. To avoid the curse of dimensionality of RBF NN, first-order filters are introduced to reduce the number of NN approximators and decrease the dimension of NN inputs. The control law is constructed based on Lyapunov stability, which guarantees the closed-loop stability and the tracking error satisfying the prescribed performance during the transient process. Subsequently, due to the property of DSC and structure features of the considered manipulator, a system decomposition strategy is employed to decompose the stable closed-loop system into two linear time-varying (LTV) perturbed subsystems on the basis of the number of NNs in the whole system. Through the recursive

design, the recurrent properties of NN input variables are easily proven. Consequently, with the satisfaction of the PE condition of RBF NNs, the convergence of partial neural weights is verified, and the unknown dynamics of system are approximated accurately in a local region along recurrent orbits. By utilization of the constant neural weights stored, a neural learning controller is developed to achieve the closed-loop stability and better control performance under the prescribed constraints for the same or similar control task. Compared with the existing neural learning results, the proposed neural learning control scheme not only achieves better control performance with specified transient and steady-state constraints but also reduces the dimension of NN inputs and the number of NNs significantly.

This paper is organized as follows. In Section 2, the problem formulation and preliminaries are stated before the control scheme design. In Section 3, a novel adaptive neural dynamic surface control scheme is proposed to guarantee that the constrained tracking error converges to a small neighborhood around zero with the prescribed performance in a finite time, and all the signals in the closed-loop system are uniformly ultimately bounded. Section 4 shows that the knowledge acquisition, expression, storage, and utilization of the manipulator's unknown dynamics can be achieved after the steady-state control process. To verify the effectiveness of the proposed control scheme, simulation results on a single-link flexible joint manipulator and experiment results on Baxter robot are given in Section 5. Last but not least, the conclusions are drawn in Section 6.

2. Problem Formulation and Preliminaries

2.1. System Formulation. In this paper, we consider an n -link manipulator with flexible joints, whose model is described by [1]

$$\begin{aligned} M(q_1) \ddot{q}_1 + C(q_1, \dot{q}_1) \dot{q}_1 + g(q_1) + K(q_1 - q_2) &= 0, \\ J \ddot{q}_2 - K(q_1 - q_2) &= u, \end{aligned} \quad (1)$$

where $q_1 \in R^n$ is the vector of links' angle positions and $q_2 \in R^n$ is the vector of motors' angle positions. $M(q_1) \in R^{n \times n}$ is the link inertia matrix and $J \in R^{n \times n}$ is the diagonal and positive definite motor inertia matrix. Moreover, $C(q_1, \dot{q}_1) \in R^{n \times n}$ denotes the Coriolis and centrifugal matrix and $g(q_1) \in R^n$ represents the gravitational terms. $K \in R^{n \times n}$ is a diagonal and positive definite matrix of joint spring constants; thus K^{-1} is also positive definite. Finally, $u \in R^n$ is the control input of system (1), and the output of system (1) is q_1 .

Property 1 (see [2]). The inertia matrix $M(q_1)$ is symmetric and positive definite; both $M(q_1)$ and $M^{-1}(q_1)$ are uniformly bounded.

Property 2 (see [2]). The Coriolis and centrifugal matrix $C(q_1, \dot{q}_1)$ can be defined such that $\dot{M}(q_1) - 2C(q_1, \dot{q}_1)$ is skew symmetric; that is, $x^T(\dot{M}(q_1) - 2C(q_1, \dot{q}_1))x = 0, \forall x \in R^n$.

The reference trajectory vector $y_d \in R^n$ is generated by the following smooth and bounded reference model:

$$\begin{aligned} \dot{x}_{di} &= x_{d,i+1}, \quad i = 1, \dots, m-1, \\ \dot{x}_{dm} &= f_d(x_d), \\ y_d &= x_{d1}, \end{aligned} \quad (2)$$

where $x_d = [x_{d1}^T, \dots, x_{dm}^T]^T \in R^{m \cdot n}$ and y_d is the system outputs vector. $f_d(\cdot)$ is a smooth known nonlinear function. Assume $x_{di} \in R^n$, $1 \leq i \leq m$ are recurrent signals and the reference orbit (denoted as $\psi_d(x_d(0))$) is a recurrent motion. Moreover, $M^{-1}(q_1)$, $C(q_1, \dot{q}_1)$, $g(q_1)$, K , and J are assumed as unknown terms.

Our goal is to design a neural learning controller, which forces the tracking error vector (i.e., $e_1 = q_1 - y_d$) converges to a small neighborhood around zero with prescribed performance in a finite time. Before the design of learning control (LC) scheme, a stable adaptive neural dynamic surface controller with prescribed performance is developed to verify the feasibility of ANC scheme. According to the deterministic learning theory, the unknown system dynamics are accurately approximated by localized RBF networks along the recurrent orbits of NN inputs. Then, based on the ANC scheme and the approximation of localized RBF networks, the knowledge on unknown system dynamics is stored in static neural weights, which is also reused to develop a neural learning controller. This neural learning controller is verified to achieve the closed-loop system stability and better control performance with prescribed constraints for the same or similar tasks.

2.2. Prescribed Tracking Performance. In this paper, the output error vector of system (1) is defined as $e_1 = q_1 - y_d = [e_{11}, e_{12}, \dots, e_{1n}]^T \in R^n$. To achieve the prescribed performance (i.e., overshoot, convergence rate, and convergence accuracy), each element in e_1 is constrained into the following prescribed region:

$$-\delta_{1k}\beta(t) < e_{1k} < \delta_{2k}\beta(t), \quad k = 1, 2, \dots, n, \quad (3)$$

where δ_{1k} and δ_{2k} are positive design constants. $\beta(t)$ is a bounded, smooth, strictly positive, and decreasing performance function. In addition, $\beta(t)$ is chosen as the following form by setting $\lim_{t \rightarrow 0} \beta(t) = \beta_0$ and $\lim_{t \rightarrow \infty} \beta(t) = \beta_\infty$:

$$\beta(t) = (\beta_0 - \beta_\infty)e^{-\kappa t} + \beta_\infty, \quad (4)$$

where β_0 , β_∞ , and κ are positive constants. With (3) and (4), it can be concluded that the convergence rate of $e_{1k}(t)$ is constrained by the decreasing rate κ of $\beta(t)$, while its maximum bound of overshoot at initial moment is constrained by $-\delta_{1k}\beta_0$ and $\delta_{2k}\beta_0$, and its steady error is constrained within a range from $-\delta_{1k}\beta_\infty$ to $\delta_{2k}\beta_\infty$.

2.3. RBF Neural Network. According to [45], RBF NN can approximate any continuous function vector $F(Z) \in R^n$ over a compact set $\Omega_Z \in R^m$ to any arbitrary accuracy as

$$F(Z) = W^{*T}S(Z) + \epsilon(Z), \quad \forall Z \in \Omega_Z \in R^m, \quad (5)$$

where $W^* \in R^{l \times n}$ is the ideal weights matrix, l is the NN node number, $S(Z) = [s_1(Z), s_2(Z), \dots, s_l(Z)]^T \in R^l$ is the basis function vector with $s_i(Z)$ ($i = 1, 2, \dots, l$) chosen as the Gaussian function $s_i(Z) = \exp[-(Z - \mu_i)^T(Z - \mu_i)/2\eta_i^2]$, and $\epsilon(Z) = [\epsilon_1(Z), \epsilon_2(Z), \dots, \epsilon_n(Z)]^T \in R^n$ is the approximation error vector which satisfies $\|\epsilon_i(Z)\| \leq \epsilon_i^*$, $i = 1, 2, \dots, n$, with constant $\epsilon_i^* > 0$.

On the other hand, it has been shown in [46] that, for any bounded trajectory $Z(t)$ over the compact set Ω_Z , the continuous and smooth function $F(Z)$ can be approximated to an arbitrary accuracy using the localized RBF NNs with a limited number of neurons located in a local region along the trajectory:

$$F(Z) = W_\zeta^{*T}S_\zeta(Z) + \epsilon_\zeta(Z), \quad (6)$$

where $S_\zeta(Z) \in R^l$ is the subvector of $S(Z)$ and $W_\zeta^{*T} \in R_\zeta^l$ with $l_\zeta < l$. $\epsilon_\zeta(Z)$ is the approximation error which is close to $\epsilon(Z)$.

Lemma 1 ((partial PE condition for RBF NNs) [46]). *Consider any continuous recurrent trajectory $Z(t)$. Assume that $Z(t)$ is a continuous map from $[0, \infty)$ into R^q , and $Z(t)$ remains in a bounded compact set Ω_Z with $\Omega_Z \subset R^q$. Then, for the RBF NN $W^T S(Z)$ with centers placed on a regular lattice (large enough to cover the compact set Ω_Z), the regression subvector $S_\zeta(Z)$ consisting of RBFs with centers located in a small neighborhood of $Z(t)$ is persistently exciting.*

3. Adaptive Neural DSC Design with Predefined Tracking Performance

In this section, performance function is introduced for describing constraints of system (1). Then an adaptive neural DSC is developed, with the design of adaptive control law based on the transformed error. Meanwhile, RBF NN is used to approximate the unknown dynamics.

Step 1. Similarly to the traditional backstepping design, we set

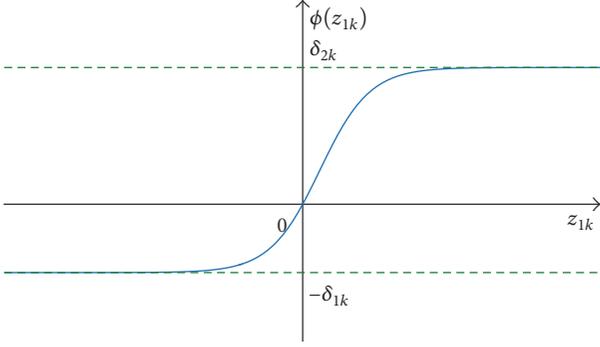
$$e_1 = q_1 - y_d = [e_{11}, e_{12}, \dots, e_{1n}]^T \in R^n. \quad (7)$$

It should be pointed out that any errors set in previously traditional design are under unrestricted condition [20], while e_1 is constrained to satisfy condition (3) in this paper, which is rewritten as

$$-\delta_{1k} < \frac{e_{1k}}{\beta(t)} < \delta_{2k}, \quad k = 1, 2, \dots, n. \quad (8)$$

It implies that e_1 can not be used for design directly due to the limitation of the traditional design method. To solve the constrained tracking control problem, the constrained error should be transformed into the unconstrained one equivalently. Therefore, a new error vector is defined as $z_1 = [z_{11}, z_{12}, \dots, z_{1n}]^T \in R^n$ called transformed error vector. Define a smooth transformed functions vector $\Phi(z_1) = [\phi(z_{11}), \phi(z_{12}), \dots, \phi(z_{1n})]^T$ with $\phi(z_{1k})$ being chosen as

$$\phi(z_{1k}) = \delta_{ak} \arctan\left(z_{1k} - \tan\left(\frac{\delta_{bk}}{\delta_{ak}}\right)\right) + \delta_{bk}, \quad (9)$$

FIGURE 1: Transformed function $\phi(z_{1k})$.

where $\delta_{ak} = (\delta_{2k} + \delta_{1k})/\pi$, and $\delta_{bk} = (\delta_{2k} - \delta_{1k})/2$. Moreover, for the symmetric tracking error constraints $-\delta_k \beta(t) < e_{1k} < \delta_k \beta(t)$, $k = 1, 2, \dots, n$, the transformation function (9) can be constructed as $\phi(z_{1k}) = (2/\pi)\delta_k \arctan(z_{1k})$. For clarity, Figure 1 illustrates the relationship between $\phi(z_{1k})$ and z_{1k} .

It can be concluded from (9) and Figure 1 that the transformation function $\phi(z_{1k})$ is smooth and strictly increasing while possessing the following properties:

$$\begin{aligned} -\delta_{1k} < \phi(z_{1k}) < \delta_{2k}, \quad \forall z_{1k} \in L_\infty, \\ \lim_{z_{1k} \rightarrow +\infty} \phi(z_{1k}) &= \delta_{2k}, \\ \lim_{z_{1k} \rightarrow -\infty} \phi(z_{1k}) &= -\delta_{1k}. \end{aligned} \quad (10)$$

By combining (8) with (10), e_{1k} can be rewritten as

$$e_{1k} = \phi(z_{1k}) \beta(t), \quad k = 1, 2, \dots, n. \quad (11)$$

Since $\phi(\cdot)$ is a strictly monotonic increasing function and $\beta(t) \neq 0$, the inverse function z_{1k} of $\phi(\cdot)$ exists, which can be expressed as

$$z_{1k} = \tan\left(\frac{e_{1k}}{\delta_{ak}\beta(t)} - \frac{\delta_{bk}}{\delta_{ak}}\right) + \tan\left(\frac{\delta_{bk}}{\delta_{ak}}\right). \quad (12)$$

Noting that $z_1 = [z_{11}, z_{12}, \dots, z_{1n}]^T$, its derivative can be presented as

$$\dot{z}_1 = \Upsilon \left(\dot{q}_1 - \dot{y}_d - \frac{\dot{\beta}(t)}{\beta(t)} e_1 \right), \quad (13)$$

where $\Upsilon = \text{diag}(\gamma_1, \gamma_2, \dots, \gamma_n)$, with γ_k ($k = 1, 2, \dots, n$) being presented as

$$\gamma_k = \frac{1}{\delta_{ak}\beta(t)} \left[1 + \tan^2\left(\frac{e_{1k}}{\delta_{ak}\beta(t)} - \frac{\delta_{bk}}{\delta_{ak}}\right) \right]. \quad (14)$$

It is clear that Υ is positive definite, which is helpful for the stability analysis. By introducing a new filter variable $\alpha_{1f} \in R^n$ and noting $z_2 = \dot{q}_1 - \alpha_{1f}$, then the virtual controller $\alpha_1 \in R^n$ is constructed as

$$\alpha_1 = -(\Upsilon^{-1}C_1 + \Upsilon)z_1 + \dot{y}_d + \frac{\dot{\beta}(t)}{\beta(t)}e_1, \quad (15)$$

where $C_1 \in R^{n \times n}$ is a diagonal and positive design matrix. Take α_1 as the input of a first-order filter and α_{1f} as the output of it, a differential equation is constructed as

$$\tau_1 \dot{\alpha}_{1f} + \alpha_{1f} = \alpha_1, \quad \alpha_{1f}(0) = \alpha_1(0), \quad (16)$$

where τ_1 is the filter time constant and set $y_1 = \alpha_{1f} - \alpha_1$; then (13) can be rewritten as

$$\dot{z}_1 = -(C_1 + \Upsilon^T \Upsilon)z_1 + \Upsilon z_2 + \Upsilon y_1. \quad (17)$$

Step 2. Let $z_2 = \dot{q}_1 - \alpha_{1f}$, its derivative can be obtained:

$$\begin{aligned} \dot{z}_2 &= M^{-1}(q_1)K(q_2 - q_1 - K^{-1}C(q_1, \dot{q}_1)\alpha_{1f} \\ &\quad - K^{-1}g(q_1) - K^{-1}M(q_1)\dot{\alpha}_{1f} - K^{-1}C(q_1, \dot{q}_1)z_2). \end{aligned} \quad (18)$$

Define the unknown dynamics in (18) as

$$\begin{aligned} H_2(\Psi_2) &= -K^{-1}C(q_1, \dot{q}_1)\alpha_{1f} - K^{-1}g(q_1) \\ &\quad - K^{-1}M(q_1)\dot{\alpha}_{1f}, \end{aligned} \quad (19)$$

where

$$\Psi_2 = [q_1^T, \dot{q}_1^T, \dot{\alpha}_{1f}^T]^T \in R^{3n}. \quad (20)$$

According to the property of RBF NN, $H_2(\Psi_2)$ can be approximated accurately by RBF NN and (19) is rewritten as

$$H_2(\Psi_2) = W_2^{*T}S_2(\Psi_2) + \epsilon_2, \quad (21)$$

where $\epsilon_2 \in R^n$ is the bounded approximation error vector which satisfies $\|\epsilon_2\| \leq \epsilon_2^*$. Define \widehat{W}_2 as the estimate of W_2^* and set $\widetilde{W}_2 = \widehat{W}_2 - W_2^*$. Then a new filter variable $\alpha_{2f} \in R^n$ is introduced and note $z_3 = z_2 - \alpha_{2f}$; (18) can be rewritten by combining (19) and (21):

$$\begin{aligned} \dot{z}_2 &= M^{-1}(q_1)K(z_3 + \alpha_{2f} - q_1 + W_2^{*T}S_2(\Psi_2) + \epsilon_2 \\ &\quad - K^{-1}C(q_1, \dot{q}_1)z_2). \end{aligned} \quad (22)$$

Then the virtual controller $\alpha_2 \in R^n$ is constructed as

$$\alpha_2 = -\Upsilon z_1 - C_2 z_2 - \widehat{W}_2^T S_2(\Psi_2) + q_1, \quad (23)$$

where $C_2 \in R^{n \times n}$ is a diagonal and positive design matrix. The updated law of NN weights is given by

$$\dot{\widehat{W}}_2 = \Gamma_2 [S_2(\Psi_2)z_2^T - \sigma_2 \widehat{W}_2] \quad (24)$$

with diagonal matrix $\Gamma_2 = \Gamma_2^T > 0$ and small value $\sigma_2 > 0$ for enhancing the robustness of the controller (23).

Take α_2 as the input of a first-order filter and α_{2f} as the output of it, a differential equation is constructed as

$$\tau_2 \dot{\alpha}_{2f} + \alpha_{2f} = \alpha_2, \quad \alpha_{2f}(0) = \alpha_2(0), \quad (25)$$

where τ_2 is the filter time constant and set $y_2 = \alpha_{2f} - \alpha_2$; then (22) can be rewritten as

$$\begin{aligned} \dot{z}_2 &= M^{-1}(q_1)K(-\Upsilon z_1 - C_2 z_2 + z_3 + y_2 \\ &\quad - \widehat{W}_2^T S_2(\Psi_2) + \epsilon_2 - K^{-1}C(q_1, \dot{q}_1)z_2). \end{aligned} \quad (26)$$

Step 3. Define $z_3 = q_2 - \alpha_{2f}$, its derivative can be obtained:

$$\dot{z}_3 = \dot{q}_2 - \dot{\alpha}_{2f}. \quad (27)$$

Introduce a new filter variable $\alpha_{3f} \in R^n$ and note $z_4 = \dot{q}_2 - \alpha_{3f}$. Then the virtual controller $\alpha_3 \in R^n$ is constructed as

$$\alpha_3 = -z_2 - C_3 z_3 + \dot{\alpha}_{2f}, \quad (28)$$

where $C_3 \in R^{n \times n}$ is a diagonal and positive design matrix. Take α_3 as the input of a first-order filter and α_{3f} as the output of it, a differential equation is constructed as

$$\tau_3 \dot{\alpha}_{3f} + \alpha_{3f} = \alpha_3, \quad \alpha_{3f}(0) = \alpha_3(0), \quad (29)$$

where τ_3 is the filter time constant and set $y_3 = \alpha_{3f} - \alpha_3$; then (27) can be rewritten as

$$\dot{z}_3 = -z_2 - C_3 z_3 + z_4 + y_3. \quad (30)$$

Step 4. Let $z_4 = \dot{q}_2 - \alpha_{3f}$; the following can be obtained:

$$\dot{z}_4 = J^{-1} [u + K(q_1 - q_2) - J\dot{\alpha}_{3f}]. \quad (31)$$

Define the unknown dynamics of the system as

$$H_4(\Psi_4) = K(q_1 - q_2) - J\dot{\alpha}_{3f}, \quad (32)$$

where

$$\Psi_4 = [q_1^T, q_2^T, \dot{\alpha}_{3f}^T]^T \in R^{3n}. \quad (33)$$

According to the property of RBF NN, $H_4(\Psi_4)$ can be approximated accurately by RBF NN and (32) is rewritten as

$$H_4(\Psi_4) = W_4^{*T} S_4(\Psi_4) + \epsilon_4, \quad (34)$$

where $W_4^* \in R^{j \times n}$ is the ideal constant weight matrix with j being the NN node number, $S_4(\Psi_4) \in R^j$ is the basis function vector, and ϵ_4 is the bounded approximation error vector which satisfies $\|\epsilon_4\| \leq \epsilon_4^*$. Define \widehat{W}_4 as the estimate of W_4^* and let $\widetilde{W}_4 = \widehat{W}_4 - W_4^*$. Then (31) can be rewritten by combining (32) and (34):

$$\dot{z}_4 = J^{-1} [u + W_4^{*T} S_4(\Psi_4) + \epsilon_4]. \quad (35)$$

Then the control input u is constructed as

$$u = -z_3 - C_4 z_4 - \widehat{W}_4^T S_4(\Psi_4), \quad (36)$$

where $C_4 \in R^{n \times n}$ is a diagonal and positive design matrix. The updated law of NN weights is given by

$$\dot{\widehat{W}}_4 = \Gamma_4 [S_4(\Psi_4) z_4^T - \sigma_4 \widehat{W}_4] \quad (37)$$

with diagonal matrix $\Gamma_4 = \Gamma_4^T > 0$ and small value $\sigma_4 > 0$ for enhancing the robustness of the controller (36). Then (35) can be rewritten as

$$\dot{z}_4 = J^{-1} [-z_3 - C_4 z_4 - \widetilde{W}_4^T S_4(\Psi_4) + \epsilon_4]. \quad (38)$$

Let us construct the following Lyapunov function candidate:

$$V = \frac{1}{2} \sum_{i=1}^4 z_i^T K_i z_i + \frac{1}{2} \sum_{k=1}^2 \text{tr} [\widetilde{W}_{2k}^T \Gamma_{2k}^{-1} \widetilde{W}_{2k}] + \frac{1}{2} \sum_{l=1}^3 y_l^T y_l, \quad (39)$$

where $K_1 = I, K_2 = K^{-1} M(q_1), K_3 = I,$ and $K_4 = J$.

Remark 2. It should be pointed out that, in the adaptive neural backstepping design [20], the derivative $\dot{\alpha}_2$ of the virtual control α_2 in Step 2 is usually used to design the virtual control α_3 in Step 3. However, according to (1) and (23), it can be seen clearly that $\dot{\alpha}_2$ is not available because the unknown terms, such as $M^{-1}(q_1), C(q_1, \dot{q}_1),$ and $g(q_1),$ are included in $\dot{\alpha}_2$. Therefore, a neural network has to be employed in Step 3 of backstepping to approximate the unknown dynamics in $\dot{\alpha}_2$. However, too many neural networks employed make the control scheme implemented difficultly. To solve this problem, this paper introduces a new variable α_{2f} in (25) to design virtual control α_2 using a first-order filter. From (25), it is easy to calculate that $\dot{\alpha}_{2f} = (\alpha_2 - \alpha_{2f})/\tau_2 = -y_2/\tau_2$. The advantage of the proposed approach is that the unknown dynamics in the previous step does not affect the design of virtual control in next step, so that the number of NNs employed can be greatly reduced. Moreover, the proposed method uses $\dot{\alpha}_{if}$, instead of the intermediate variables used in [20], as neural input variable, so that there are only $3n$ neural input variables for neural networks $W_2^{*T} S_2(\Psi_2)$ and $W_4^{*T} S_4(\Psi_4)$, respectively, which reduces significantly the number of neural input variables used in [20], where $4n$ and $8n$ neural input variables are used in $W_2^{*T} S_2(\Psi_2)$ and $W_4^{*T} S_4(\Psi_4)$.

Theorem 3. Consider the manipulator model (1), the reference trajectory y_d , the prescribed performance bounds (8), the state transformation (11), the adaptive NN control law (36), the weight updated law (24), (37), and Lyapunov function (39). Given any constant $\mu > 0$, for any bounded initial conditions satisfying the prescribed performance (8) and $V(0) < \mu$, there exists design parameters $C_1, C_2, C_3, C_4, \Gamma_2, \sigma_2, \Gamma_4, \sigma_4, \tau_1, \tau_2,$ and τ_3 , such that the proposed control scheme guarantees that (1) all the signals in the closed-loop system are uniformly ultimately bounded and (2) the tracking error converges to a small neighborhood around zero with the prescribed performance (8) in a finite time T_1 .

Proof. See Appendix A. \square

4. Dynamic Neural Learning

In this section, we will show the learning ability of RBF NNs for unknown system dynamics $H_2(\Psi_2)$ and $H_4(\Psi_4)$ in the case of the manipulator with predefined tracking performance. Subsequently, the stored knowledge on the unknown dynamics will be utilized to design a neural learning controller achieving better control performance, while satisfying the prescribed performance.

4.1. Learning from Adaptive Neural DSC. According to Lemma 1, it can be concluded that a recurrent orbit can make

regression subvectors $S_\zeta(Z)$ satisfy the partial PE condition, which is a key condition to ensure the accurate convergence of neural weights.

Theorem 4. Consider the closed-loop system consisting of the flexible joint manipulator model (1), the reference trajectory y_d , the prescribed performance bounds (8), the state transformation (11), the adaptive NN control law (36), and the weight updated law (24) and (37). Then, for any recurrent orbit $\psi_d(x_d(t))|_{t \geq 0}$ and initial condition $x(0) \in \Omega_0$ (Ω_0 is an appropriately chosen compact set) satisfying the prescribed performance (8) and $\widehat{W}_2(0) = \widehat{W}_4(0) = 0$, we have that the neural weights \widehat{W}_i converge to small neighborhoods around optimal values W_i^* , and the locally accurate approximation of the system dynamics $H_i(\Psi_i)$ is obtained by the stored knowledge \overline{W}_i :

$$\overline{W}_i = \text{mean}_{t \in [t_{a_i}, t_{b_i}]} \widehat{W}_i(t) \quad (40)$$

with $i = 2, 4$, $T_1 \leq t_{a_2} \leq t_{b_2} \leq T_2 \leq t_{a_4} \leq t_{b_4}$. $[t_{a_i}, t_{b_i}]$ and $[t_{a_1}, t_{b_1}]$ are time segments after the steady-state control process.

Proof. From Theorem 3, all the signals in the closed-loop system are uniformly ultimately bounded and the tracking error vector $e_1 = q_1 - y_d$ converges to a small neighborhood around zero with the prescribed performance in a finite time T_1 . Thus, the state q_1 converges closely to the recurrent signals y_d for all $t \geq T_1$. In addition, it can be obtained from the proof of Theorem 3 that the transformed error vector z_1 converges exponentially to a small neighborhood around zero in a finite time T_1 . From (15), it can be concluded that virtual control α_1 is recurrent with the same period as \dot{y}_d by combining the convergence of z_1 and e_1 . Noting $\dot{q}_1 = z_2 + y_1 + \alpha_1$ with z_2 and y_1 being close to a small neighborhood around zero based on Theorem 3, then \dot{q}_1 is a recurrent signal with the same period as α_1 . In addition, α_{1f} is also a recurrent signal with the same period as α_1 since $y_1 = \alpha_{1f} - \alpha_1$ and y_1 is a small value. From (16), it can be obtained that $\dot{\alpha}_{1f}$ is a recurrent signal as well. Therefore, the NN inputs $\Psi_2 = [q_1^T, \dot{q}_1^T, \dot{\alpha}_{1f}^T]^T$ are recurrent for all $t \geq T_1$, and a partial PE condition of $S_2(\Psi_2)$ is satisfied according to Lemma 1. By combining the convergence of z_1 and the localized RBF NN along the recurrent signals $\Psi_2(t)$ ($t > T_1$), it can be obtained from (24) and (26) that

$$\dot{z}_2 = M^{-1}(q_1)K \left[-Yz_1 - C_2z_2 + z_3 + y_2 - \overline{W}_{2\zeta}^T S_{2\zeta}(\Psi_2) + \epsilon_{2\zeta} - K^{-1}C(q_1, \dot{q}_1)z_2 \right] \quad (41)$$

$$\begin{aligned} \dot{\overline{W}}_{2\zeta} &= \Gamma_{2\zeta} \left[S_{2\zeta}(\Psi_2)z_2^T - \sigma_2 \overline{W}_{2\zeta} \right], \\ \dot{\overline{W}}_{\overline{2\zeta}} &= \Gamma_{\overline{2\zeta}} \left[S_{\overline{2\zeta}}(\Psi_2)z_2^T - \sigma_2 \overline{W}_{\overline{2\zeta}} \right], \end{aligned} \quad (42)$$

where the subscript ζ stands for the region near the orbits $\Psi_2(t)$, $S_{2\zeta}(\Psi_2)$ is the subvector of $S_2(\Psi_2)$ consisting of the corresponding RBFs, and $\overline{W}_{2\zeta}$ is the corresponding weight submatrix of \overline{W}_2 . Moreover, the subscript $\overline{\zeta}$ represents the region

away from the orbits $\Psi_2(t)$, and $\epsilon_{2\zeta} = \epsilon_2(\Psi_2) - \overline{W}_{2\zeta}^T S_{2\zeta}(\Psi_2)$ is the NN approximation error along the orbits $\Psi_2(t)$. Since $\|\overline{W}_{2\zeta}^T S_{2\zeta}(\Psi_2)\|$ is small, $\|\epsilon_{2\zeta}\|$ is close to $\|\epsilon_2(\Psi_2)\|$. \square

It would be shown that the perturbation term $M^{-1}(q_1)K\epsilon_{2\zeta}$ may be large, which will make the accurate convergence of neural weights become difficult. To solve this problem, a state transformation $z_{2s} = K^{-1}M(q_1)z_2$ is introduced to eliminate the influence of $M^{-1}(q_1)K\epsilon_{2\zeta}$. Subsequently, set $\overline{W}_{2\zeta} = [\overline{W}_{21\zeta}, \overline{W}_{22\zeta}, \dots, \overline{W}_{2n\zeta}] \in R^{\zeta \times n}$ and $\overline{W}_{2\overline{\zeta}} = [\overline{W}_{21\overline{\zeta}}, \overline{W}_{22\overline{\zeta}}, \dots, \overline{W}_{2n\overline{\zeta}}] \in R^{\zeta \times n}$; (41) can be transformed into the following form:

$$\begin{aligned} \begin{bmatrix} \dot{z}_{2s} \\ \dot{\overline{W}}_{21\zeta} \\ \vdots \\ \dot{\overline{W}}_{2n\zeta} \end{bmatrix} &= \begin{bmatrix} A_2(t) & B_2(t) \\ C_2(t) & 0 \end{bmatrix} \begin{bmatrix} z_{2s} \\ \overline{W}_{21\zeta} \\ \vdots \\ \overline{W}_{2n\zeta} \end{bmatrix} \\ &+ \begin{bmatrix} \epsilon'_{2\zeta} \\ -\sigma_2 \Gamma_{2\zeta} \overline{W}_{21\zeta} \\ \vdots \\ -\sigma_2 \Gamma_{2\zeta} \overline{W}_{2n\zeta} \end{bmatrix}, \end{aligned} \quad (43)$$

where

$$\begin{aligned} A_2(t) &= -FM^{-1}(q_1)K \in R^{n \times n} \\ B_2(t) &= -\text{diag}\{S_{2\zeta}^T, \dots, S_{2\zeta}^T\} \in R^{n \times \zeta n} \\ C_2(t) &= \overline{\Gamma}_{2\zeta} \overline{S}_{2\zeta} M^{-1}(q_1)K \in R^{\zeta \times n \times n} \\ F &= C_2 - K^{-1}[\dot{M}(q_1) - C(q_1, \dot{q}_1)] \in R^{n \times n} \\ \overline{\Gamma}_{2\zeta} &= \text{diag}\{\Gamma_{2\zeta}, \dots, \Gamma_{2\zeta}\} \in R^{\zeta n \times \zeta n} \\ \overline{S}_{2\zeta} &= \text{diag}\{S_{2\zeta}, \dots, S_{2\zeta}\} \in R^{\zeta n \times n} \\ \epsilon'_{2\zeta} &= Yz_1 + z_3 + y_2 + \epsilon_{2\zeta}. \end{aligned} \quad (44)$$

According to Theorem 3, $\epsilon'_{2\zeta}$ is a small value and $-\sigma_2 \Gamma_{2\zeta} \overline{W}_{2i\zeta}$ ($i = 1, 2, \dots, n$) can be made as a small value by choosing a small design parameter σ_2 . Therefore, system (43) can be considered as a linear time-varying (LTV) system with a small perturbation term. Choose $P(t) = M^{-1}(q_1)K$; then

$$\begin{aligned} \dot{P}(t) + P(t)A_2(t) + A_2^T(t)P(t) \\ = \dot{M}^{-1}(q_1)K - (F_1 C_2 + F_2)M^{-1}(q_1)K, \end{aligned} \quad (45)$$

where $F_1 = M^{-1}(q_1)K + KM^{-1}(q_1)$, $F_2 = KM^{-1}(q_1)[\dot{M}(q_1) - C(q_1, \dot{q}_1)]K^{-1} - M^{-1}(q_1)[\dot{M}(q_1) - C(q_1, \dot{q}_1)]$. Choose C_2 appropriately such that $\dot{P}(t) + P(t)A_2(t) + A_2^T(t)P(t) < 0$. Subsequently, based on the perturbation theory from Lemma 4.6 in [47], both z_{2s} and $\overline{W}_{2\zeta}$ converge exponentially to a small

neighborhood around zero in a finite time T_1 , and the size of neighborhood is determined by $\|\dot{\epsilon}_{2\zeta}\|$ and $\|\sigma_2 \Gamma_{2\zeta} \widehat{W}_{2\zeta}\|$, respectively. Noting $\widehat{W}_{2\zeta} = \widehat{W}_{2\zeta} - W_{2\zeta}^*$, it is clear that $\widehat{W}_{2\zeta}$ can converge to a small neighborhood of optimal weights $W_{2\zeta}^*$ in a finite time T_1 , and the constant weights \overline{W}_2 can be obtained from (40). According to the localization property of RBF NN, the system dynamics $H_2(\Psi_2)$ can be described by

$$\begin{aligned} H_2(\Psi_2) &= \widehat{W}_{2\zeta}^T S_{2\zeta}(\Psi_2) + \epsilon_{2\zeta}(\Psi_2) \\ &= \overline{W}_2^T S_2(\Psi_2) + \bar{\epsilon}_2(\Psi_2), \end{aligned} \quad (46)$$

where both $\epsilon_{2\zeta}(\Psi_2)$ and $\bar{\epsilon}_2(\Psi_2)$ are close to $\epsilon_2(\Psi_2)$ due to the convergence of $\widehat{W}_{2\zeta}$.

According to the above analysis and noting $\alpha_2 = -\Upsilon z_1 - C_2 z_2 - \overline{W}_2^T S_2(\Psi_2) + q_1$, there exists a constant $T_2 > T_1$, such that the virtual control α_2 can be rewritten as

$$\alpha_2 = -\Upsilon z_1 - C_2 z_2 - \overline{W}_2^T S_2(\Psi_2) + q_1 + \epsilon_2 \quad (47)$$

for all $t > T_2$, where $\epsilon_2 = [\overline{W}_2 - \widehat{W}_2]^T S_2(\Psi_2)$ is a small value because of the convergence of \widehat{W}_2 . According to Theorem 3, α_2 is a recurrent signal. Since $y_2 = \alpha_{2f} - \alpha_2$ and y_2 is a small value, α_{2f} is also recurrent with the same period as α_2 . From (25), it can be obtained that $\dot{\alpha}_{2f}$ is also a recurrent signal as well. From (28), by combining the convergence of z_2 and z_3 , virtual control α_3 is a recurrent signal with the same period as $\dot{\alpha}_{2f}$. Since $y_3 = \alpha_{3f} - \alpha_3$ and y_3 is a small value, α_{3f} is also recurrent with the same period as α_3 . From (29), it can be obtained that $\dot{\alpha}_{3f}$ is also a recurrent signal as well. Therefore, the NN inputs $\Psi_4 = [q_1^T, q_2^T, \dot{\alpha}_{3f}^T]^T$ are recurrent for all $t \geq T_2$, and a partial PE condition of $S_4(\Psi_4)$ is satisfied according to Lemma 1. By combining the convergence of z_4 and the localized RBF NN along the recurrent signals $\Psi_4(t)$ ($t > T_2$), it can be obtained from (37) and (38) that

$$\begin{aligned} \dot{z}_4 &= J^{-1} [-z_3 - C_4 z_4 - \widehat{W}_{4\zeta}^T S_{4\zeta}(\Psi_4) + \epsilon_{4\zeta}] \\ \dot{\widehat{W}}_{4\zeta} &= \Gamma_{4\zeta} [S_{4\zeta}(\Psi_4) z_4^T - \sigma_4 \widehat{W}_{4\zeta}], \\ \dot{\widehat{W}}_{4\bar{\zeta}} &= \Gamma_{4\bar{\zeta}} [S_{4\bar{\zeta}}(\Psi_4) z_4^T - \sigma_4 \widehat{W}_{4\bar{\zeta}}]. \end{aligned} \quad (48)$$

Similarly, $S_{4\zeta}(\Psi_4)$ is the subvector of $S_4(\Psi_4)$ consisting of the RBFs near the orbits $\Psi_4(t)$, and $\widehat{W}_{4\zeta}$ is the corresponding weight submatrix of \widehat{W}_4 . Moreover, $\epsilon_{4\zeta} = \epsilon_4(\Psi_4) - \widehat{W}_{4\zeta}^T S_{4\zeta}(\Psi_4)$ is the NN approximation error along the orbits $\Psi_4(t)$. Since $\|\widehat{W}_{4\zeta}^T S_{4\zeta}(\Psi_4)\|$ is small, $\|\epsilon_{4\zeta}\|$ is close to $\|\epsilon_4(\Psi_4)\|$.

It would be shown that the perturbation term $J^{-1} \epsilon_{4\zeta}$ may be large, which will make the accurate convergence of neural weights become difficult. To solve this problem, a state transformation $z_{4s} = J z_4$ is introduced to eliminate the influence

of $J^{-1} \epsilon_{4\zeta}$. Subsequently, set $\widehat{W}_{4\zeta} = [\widehat{W}_{41\zeta}, \widehat{W}_{42\zeta}, \dots, \widehat{W}_{4n\zeta}] \in R^{\zeta' \times n}$ and $\widehat{W}_{4\bar{\zeta}} = [\widehat{W}_{41\bar{\zeta}}, \widehat{W}_{42\bar{\zeta}}, \dots, \widehat{W}_{4n\bar{\zeta}}] \in R^{\zeta' \times n}$; (46) can be transformed into the following form:

$$\begin{aligned} \begin{bmatrix} \dot{z}_{4s} \\ \dot{\widehat{W}}_{41\zeta} \\ \vdots \\ \dot{\widehat{W}}_{4n\zeta} \end{bmatrix} &= \begin{bmatrix} A_4(t) & B_4(t) \\ C_4(t) & 0 \end{bmatrix} \begin{bmatrix} z_{4s} \\ \widehat{W}_{41\zeta} \\ \vdots \\ \widehat{W}_{4n\zeta} \end{bmatrix} \\ &+ \begin{bmatrix} \epsilon'_{4\zeta} \\ -\sigma_4 \Gamma_{4\zeta} \widehat{W}_{41\zeta} \\ \vdots \\ -\sigma_4 \Gamma_{4\zeta} \widehat{W}_{4n\zeta} \end{bmatrix}, \end{aligned} \quad (49)$$

where

$$\begin{aligned} A_4(t) &= -C_4 J^{-1} \in R^{n \times n} \\ B_4(t) &= -\text{diag}\{S_{4\zeta}^T, \dots, S_{4\zeta}^T\} \in R^{n \times \zeta' n} \\ C_4(t) &= \bar{\Gamma}_{4\zeta} \bar{S}_{4\zeta} J^{-1} \in R^{\zeta' n \times n} \\ \bar{\Gamma}_{4\zeta} &= \text{diag}\{\Gamma_{4\zeta}, \dots, \Gamma_{4\zeta}\} \in R^{\zeta' n \times \zeta' n} \\ \bar{S}_{4\zeta} &= \text{diag}\{S_{4\zeta}, \dots, S_{4\zeta}\} \in R^{\zeta' n \times n} \\ \epsilon'_{4\zeta} &= -z_3 + \epsilon_{4\zeta}. \end{aligned} \quad (50)$$

Using the similar step and choosing $P'(t) = J$, we have

$$P'(t) + P'(t) A_4(t) + A_4^T(t) P'(t) = -2J C_4 J^{-1} < 0 \quad (51)$$

with $C_4 > 0$. Then, it can be proven that $\widehat{W}_{4\zeta}$ converges to a small neighborhood of optimal weights $W_{4\zeta}^*$ in a finite time T_2 , and the constant weights \overline{W}_4 can be obtained from (40). According to the localization property of RBF NN, the system dynamics $H_4(\Psi_4)$ can be described by

$$H_4(\Psi_4) = \overline{W}_4^T S_4(\Psi_4) + \bar{\epsilon}_4(\Psi_4), \quad (52)$$

where $\bar{\epsilon}_4(\Psi_4)$ is close to $\epsilon_4(\Psi_4)$ due to the convergence of $\widehat{W}_{4\zeta}$. Therefore, the dynamics $H_i(\Psi_i)$, $i = 1, 2$, can be accurately approximated by the constant RBF NN $\overline{W}_i^T S_i(\Psi_i)$ with the stored knowledge \overline{W}_i obtained in (40).

4.2. Neural Learning Control Using the Stored Knowledge. Since the locally accurate NN approximation can be achieved by the constant RBF NN $\overline{W}_i^T S_i(\Psi_i)$, for a similar control task, we will reuse the knowledge \overline{W}_i to design a neural learning controller for system (1):

$$u = -z_3 - C_4 z_4 - \overline{W}_4^T S_4(\Psi_4) \quad (53)$$

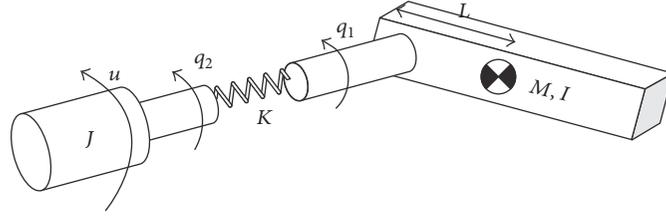


FIGURE 3: Single-link flexible joint manipulator.

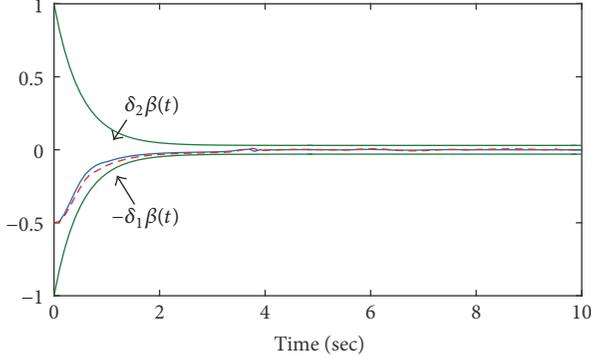


FIGURE 4: Tracking error: LC (-) and ANC (- -).

$g = 9.8$, $I = ML^2$, $J = 0.5$, and $K = 15$. Then the desired reference trajectory y_d is generated from the following equation:

$$2\ddot{y}_d + 3\dot{y}_d + y_d = 4 \sin(1.5t). \quad (57)$$

5.1. Numerical Simulation Results. Based on Theorem 3, the adaptive control law is chosen as (36), and the virtual controllers are chosen as (15), (23), and (28) with the virtual control laws (24) and (37). In the performance function, $\beta_0 = 1$, $\beta_\infty = 0.05$, $\kappa = 2$, and $\delta_1 = \delta_2 = 1$. $\widehat{W}_2^T S_2(\Psi_2)$ is constructed by 121 neurons whose centers evenly spaced on $[-0.8, 0.8] \times [-2, 2]$ and widths $\eta_{21} = 0.2$, $\eta_{22} = 0.5$. Similarly, $\widehat{W}_4^T S_4(\Psi_4)$ is constructed by 2783 neurons whose centers evenly spaced on $[-0.8, 0.8] \times [-2, 2] \times [-11, 11]$ and widths $\eta_{41} = 0.2$, $\eta_{42} = 0.5$, $\eta_{43} = 1.25$. Other design parameters are $c_1 = 2$, $c_2 = 5$, $c_3 = 2$, $c_4 = 30$, $\Gamma_2 = 5$, $\Gamma_4 = 60$, and $\sigma_2 = \sigma_4 = 0.0001$. The initial states are $[q_1(0), \dot{q}_1(0), q_2(0), \dot{q}_2(0)]^T = [0.5, 0, 0.2, 0]^T$ and $[y_d(0), \dot{y}_d(0)]^T = [1, 0]^T$. The initial weights $\widehat{W}_2(0)$ and $\widehat{W}_4(0)$ are both set as zero vectors.

According to (40), the constant neural weights \overline{W}_2 and \overline{W}_4 in (53) and (54) are obtained as $\overline{W}_2 = \text{mean}_{t \in [200, 300]} \widehat{W}_2(t)$ and $\overline{W}_4 = \text{mean}_{t \in [400, 500]} \widehat{W}_4(t)$. In order to be compared with the adaptive neural DSC results, in this simulation, the parameters in performance function and the initial states are set the same as the values set in adaptive neural DSC, while the control gain are set as $c_1 = 1$, $c_2 = 4$, $c_3 = 2$, and $c_4 = 5$.

The related simulation results are shown in Figures 4–10. Figure 4 illustrates that, for the proposed adaptive neural DSC scheme, the tracking error e_1 can ultimately

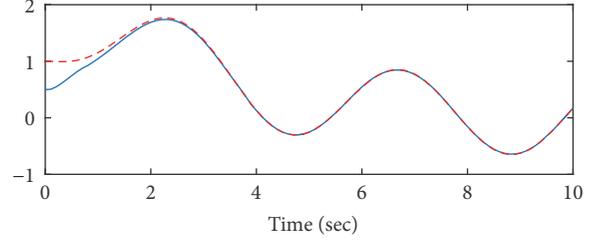
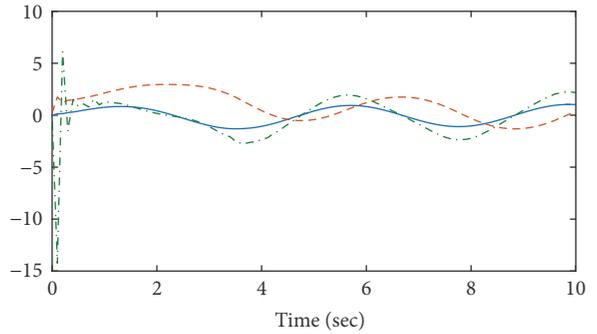
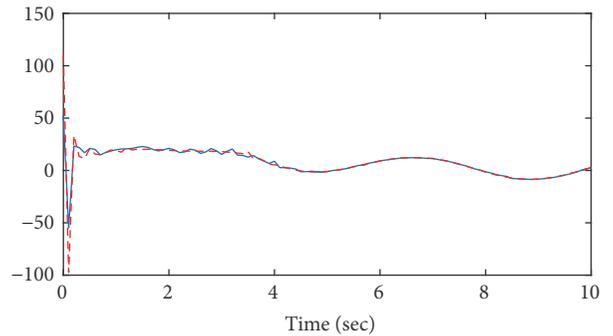
FIGURE 5: System output: x_1 (-) and y_d (- -).FIGURE 6: State variables x_2 (-), x_3 (- -), x_4 (-.-).

FIGURE 7: Control input: LC (-) and ANC (- -).

converge to a small neighborhood of zero besides satisfying the prescribed performance; for the proposed learning control scheme, not only does the tracking error satisfy the prescribed performance, but also the convergence rate is faster under the similar control input amplitude shown in Figure 7. Moreover, the time consumption is decreased by 2/3 in contrast with adaptive neural DSC scheme. Figure 5 shows that the output x_1 of system (56) tracks to the reference

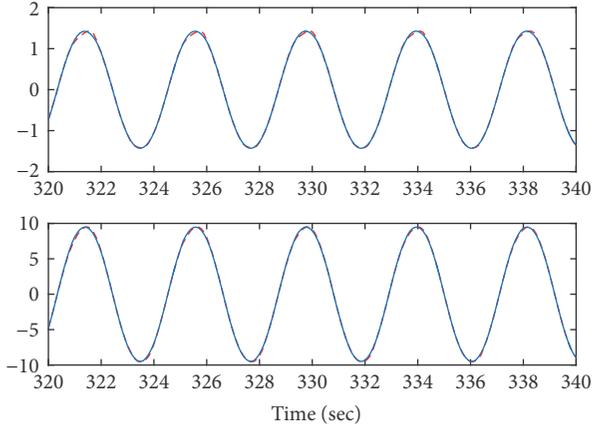


FIGURE 8: Function approximation: $\widehat{W}_i^T S_i(\Psi_i)$ (- -) and $h_i(\Psi_i)$ (-) ($i = 2, 4$).

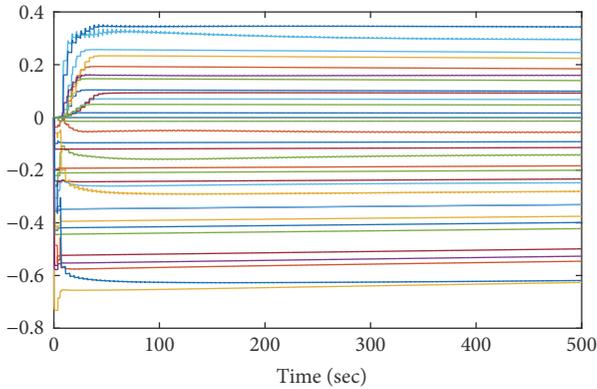


FIGURE 9: The convergence of partial NN weights \widehat{W}_2 .

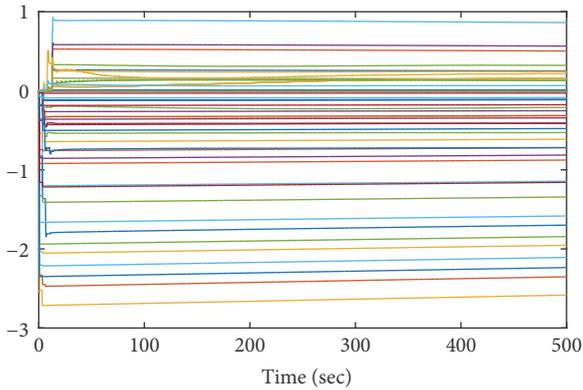


FIGURE 10: The convergence of partial NN weights \widehat{W}_4 .

trajectory y_d quickly. Figure 6 shows that other state variables are bounded for the proposed scheme. Figure 8 gives the NN approximation ability for the unknown dynamics. Figures 9 and 10 show that the NN weights converge to certain values along with the updated laws of NN weights.

Remark 7. In order to compare with the difference of tracking performance between different parameter selection in (3),

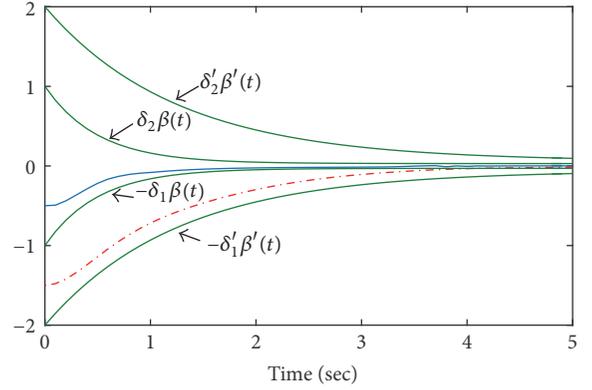


FIGURE 11: Tracking error: e (-) and e' (-.-).

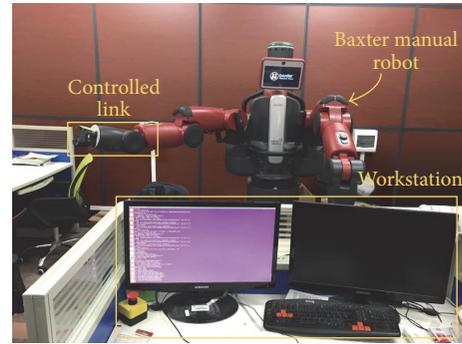


FIGURE 12: Overview of the experimental platform.

these parameters in (3) are chosen as $\beta_0 = 1$, $\beta_\infty = 0.05$, $\kappa = 2$, and $\delta_1 = \delta_2 = 1$; the others are set as $\beta'_0 = 1$, $\beta'_\infty = 0.05$, $\kappa' = 1$, and $\delta'_1 = \delta'_2 = 2$. Define the tracking error of the former as e and the latter one as e' . Figure 11 illustrates two different tracking performances of the system's output by selecting different parameters of constraints. It is evident that the tracking performance is affected by the parameter selection of performance function. And as the range of permissible error gets smaller, the tracking error is forced to converge faster and the tracking accuracy gets higher.

5.2. Experiment Results on Baxter Robot. Moreover, in order to validate the effectiveness of the proposed control scheme, the Baxter bimanual robot is used in the experiment, as shown in Figure 12. It is of two 7-DOF arms and advanced sensing technologies, including position, force, and torque sensors and control at every joint. The resolution for the joint sensors is 14 bits with 360 degrees (0.022 degrees per tick resolution), while the maximum joint torques that can be applied to the joints are 50 Nm (the first four joints) and 15 Nm (the last three joints). To compare with the simulation's results, the desired reference trajectory in the experiment is the same as one in the simulation, which is also generated from (57). In the experiment, one of the Baxter robot's links (such as the wrist's link of robot's right arm) is commanded to track the desired reference trajectory, and the tracking

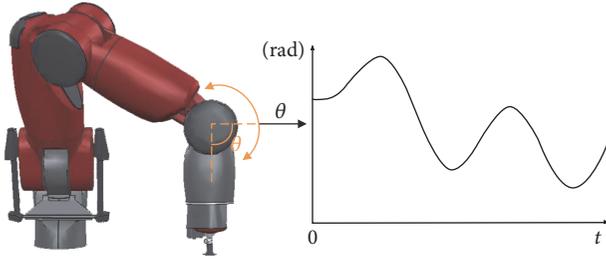


FIGURE 13: The desired motion of the robot's link.

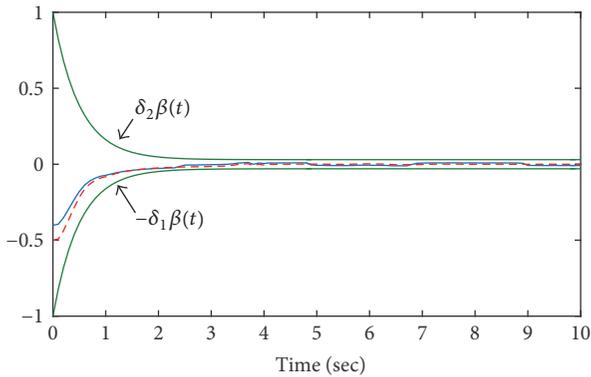


FIGURE 14: Tracking error: LC (-) of Baxter robot and LC (- -) of simulation.

error between link's angle position and reference trajectory is forced to converge to a small neighborhood around zero with prescribed performance in a finite time, while other links of the robot stay still. Figure 13 illustrates the desired motion of the robot's link.

To verify the effectiveness of neural learning control scheme for Baxter robot, the constant weights \bar{W}_2 and \bar{W}_4 in Section 5.1 are reused in experiment. For comparison, the parameters in performance function and the initial states are set the same as these values in neural learning control, while the control gains are set as $c_1 = 1.5$, $c_2 = 3$, $c_3 = 2$, and $c_4 = 4$.

The related results are shown in Figures 14–16. Figure 14 shows the difference between LC of simulation and LC of Baxter robot. Although the vibration of the robot affects the tracking performance, it is evident that the tracking error of Baxter robot can also ultimately converge to a small neighborhood of zero besides satisfying the prescribed performance. Figure 15 show that the output of Baxter robot's link (i.e., q_1 of system (56)) tracks to the reference trajectory y_d quickly, and other state variables are bounded for the proposed learning control scheme. Figure 16 shows that the control input of Baxter robot is also bounded with small overshoot and small mechanical vibration.

6. Conclusion

In this paper, we studied learning from adaptive neural dynamic surface control for a class of flexible joint manipulator with unknown dynamics under the prescribed constraint.

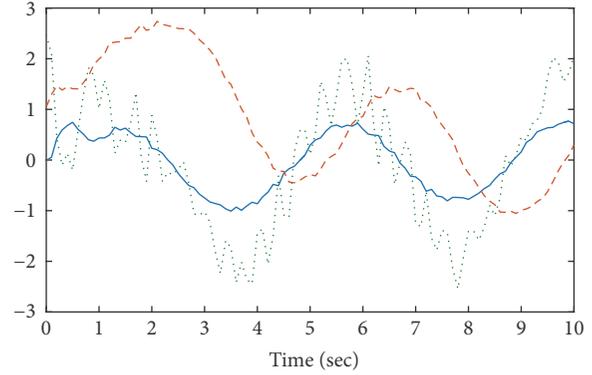
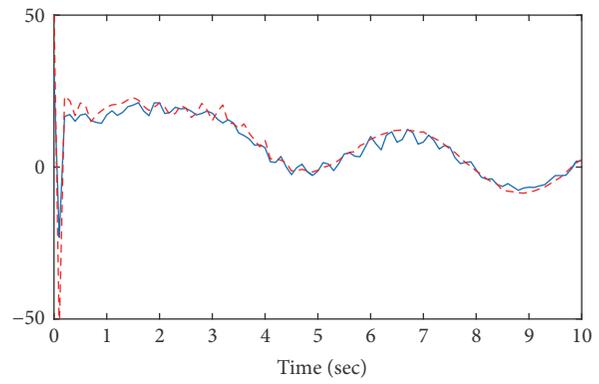
FIGURE 15: State variables q_1 (-), q_2 (- -), and \dot{q}_2 (- · -) of Baxter robot.

FIGURE 16: Control input: LC (-) of Baxter robot and LC (- -) of simulation.

A novel error transformed function was utilized to transform the constrained tracking problem into the equivalent unconstrained one so as to facilitate the controller design. Furthermore, by combining DSC method, which was used to reduce the number of NN approximators and decrease the dimension of NN inputs, a novel adaptive neural control scheme was proposed to guarantee the prescribed performance during the transient process. Then, the closed-loop stability and control performance were achieved according to the construction of the Lyapunov function. After the stable control process, since two NNs were used in the controller design, the recurrent property of the NN input variables and the partial PE condition of RBF NNs were proved recursively. Therefore, the locally accurate approximations of unknown system dynamics by RBF NNs were achieved, and the proposed control scheme was verified to be capable of storing the learned knowledge in constant RBF NNs. Finally, the stored knowledge was reused to develop the neural learning controller for the same system model and the same or similar control task, so that the closed-loop stability and better control performance were achieved under the prescribed constraint. Simulation results for a single-link flexible joint manipulator and experiment results for Baxter robot were presented to prove the effectiveness of the proposed control scheme.

Appendix

A. Proof of Theorem 3

Firstly, the derivative of y_1 , y_2 , and y_3 can be expressed as

$$\dot{y}_l = \dot{a}_{lf} - \dot{a}_l = -\frac{y_l}{\tau_l} + D_l(\cdot), \quad l = 1, 2, 3, \quad (\text{A.1})$$

where $D_1(\cdot) := D_1(z_1, z_2, y_1, y_d, \dot{y}_d, \ddot{y}_d) = C_1 \dot{z}_1 - \dot{y}_d - \dot{\beta}(t)/\beta(t)e_1 - \dot{\beta}(t)/\beta(t)[\dot{e}_1 - \dot{\beta}(t)/\beta(t)e_1]$. Similarly, $D_2(\cdot) := D_2(z_1, z_2, z_3, y_1, y_2, \widehat{W}_2, y_d, \dot{y}_d, \ddot{y}_d)$, $D_3(\cdot) := D_3(z_1, z_2, z_3, z_4, y_1, y_2, y_3, \widehat{W}_2, y_d, \dot{y}_d, \ddot{y}_d)$, and they are continuous functions. \dot{V} can be expressed as

$$\begin{aligned} \dot{V} = & -\sum_{i=1}^4 z_i^T K_i^T z_i + z_1^T Y y_1 + \sum_{j=2}^3 z_j^T y_j + z_2^T \epsilon_2 + z_4^T \epsilon_4 \\ & - \sum_{k=1}^2 \text{tr} \left[\sigma_{2k} \widehat{W}_{2k}^T \widehat{W}_{2k} \right] \\ & + \sum_{l=1}^3 \left[-\frac{y_l^T y_l}{\tau_l} + y_l^T D_l(\cdot) \right], \end{aligned} \quad (\text{A.2})$$

where $K_1' = C_1 + Y^T Y$, $K_2' = C_2 + K^{-1} C(q_1, \dot{q}_1) - K^{-1} \dot{M}(q_1)/2$, $K_3' = C_3$, $K_4' = C_4$. From $\widehat{W}_{2k} = \widehat{W}_{2k} - W_{2k}^*$ ($k = 1, 2$), the following inequality can be obtained:

$$\begin{aligned} -\text{tr} \left[\sigma_{2k} \widehat{W}_{2k}^T \widehat{W}_{2k} \right] \leq & -\frac{1}{2} \text{tr} \left[\sigma_{2k} \widehat{W}_{2k}^T \widehat{W}_{2k} \right] \\ & + \frac{1}{2} \text{tr} \left[\sigma_{2k} W_{2k}^{*T} W_{2k}^* \right]. \end{aligned} \quad (\text{A.3})$$

In addition, since y_d , \dot{y}_d , and \ddot{y}_d are recurrent signals, there exists a compact set $\Omega_d := \{(y_d, \dot{y}_d, \ddot{y}_d) : y_d^2 + \dot{y}_d^2 + \ddot{y}_d^2 \leq B_0\}$ with $B_0 > 0$. Furthermore, the set $\Omega := \{V \leq \mu\}$ is also compact by function (39), which implies that $\Omega_d \times \Omega$ is compact as well. Consequently, $D_1(\cdot)$, $D_2(\cdot)$, and $D_3(\cdot)$ are in the compact set $\Omega_d \times \Omega$. Therefore, there exists a maximum constant N_l for $D_l(\cdot)$ on $\Omega_d \times \Omega$ such that $\|D_l(\cdot)\| \leq N_l$, $l = 1, 2, 3$. According to Young's inequality, the following inequalities are obtained:

$$\begin{aligned} z_1^T Y y_1 & \leq z_1^T Y Y^T z_1 + \frac{y_1^T y_1}{4} \\ z_j^T y_j & \leq z_j^T z_j + \frac{y_j^T y_j}{4}, \quad j = 2, 3 \\ z_k^T \epsilon_k & \leq z_k^T z_k + \frac{\epsilon_k^{*2}}{4}, \quad k = 2, 4 \\ y_l^T D_l(\cdot) & \leq \frac{y_l^T y_l}{(4\rho)} + N_l^2 \rho, \quad l = 1, 2, 3, \end{aligned} \quad (\text{A.4})$$

where ρ is a positive design parameter. Let $C_2 > 2I$, $C_3 > I$, and $C_4 > I$. Then the inequality for \dot{V} can be obtained:

$$\begin{aligned} \dot{V} \leq & -\sum_{i=1}^4 c_i z_i^T K_i z_i - \sum_{k=1}^2 \text{tr} \left[\frac{\sigma_{2k}}{2} \widehat{W}_{2k}^T \widehat{W}_{2k} \right] - \sum_{l=1}^3 \xi_l y_l^T y_l \\ & + \sum_{k=1}^2 \text{tr} \left[\frac{\sigma_{2k}}{2} W_{2k}^{*T} W_{2k}^* \right] + (N_1^2 + N_2^2 + N_3^2) \rho \\ & + \frac{\epsilon_2^{*2}}{4} + \frac{\epsilon_4^{*2}}{4}, \end{aligned} \quad (\text{A.5})$$

where $c_1 = \lambda_{\min}(C_1)$, $c_2 = \lambda_{\min}[(C_2 - 2I)M^{-1}(q_1)K]$, $c_3 = \lambda_{\min}(C_3 - I)$, $c_4 = \lambda_{\min}[(C_4 - I)J^{-1}]$, $\xi_l = 1/\tau_l - N_l^2/(4\rho) - 1/4$ ($l = 1, 2, 3$). Choose τ_l ($l = 1, 2, 3$) appropriately such that $\xi_l > 0$. Moreover, set values of $C_1, C_2, C_3, C_4, \Gamma_2, \sigma_2, \Gamma_4, \sigma_4$ appropriately and (A.5) can be rewritten as

$$\dot{V} \leq -2k_0 V + b_0, \quad (\text{A.6})$$

where $b_0 = \text{tr}[\sigma_2 W_2^{*T} W_2^*]/2 + \text{tr}[\sigma_4 W_4^{*T} W_4^*]/2 + (N_1^2 + N_2^2 + N_3^2)\rho + \epsilon_2^{*2}/4 + \epsilon_4^{*2}/4$ and $k_0 = \min\{c_1, c_2, c_3, c_4, \sigma_2 \lambda_{\min}(\Gamma_2)/2, \sigma_4 \lambda_{\min}(\Gamma_4)/2, \xi_1, \xi_2, \xi_3\} > 0$. Let $\nu = b_0/(2k_0)$; then

$$V \leq [V(0) - \nu] e^{-2k_0 t} + \nu \quad (\text{A.7})$$

which means that, as $t \rightarrow \infty$, $V \leq \nu$, thus

$$\|z_i\| \leq \sqrt{\frac{2\nu}{\lambda_{\min}(K_i)}}, \quad i = 1, 2, 3, 4. \quad (\text{A.8})$$

Since ν can be made arbitrarily small by choosing appropriate design parameters k_0, ρ, σ_2 , and σ_4 . Thus, for given $\nu' > \nu$, there exists a finite time T_1 so that $\|z_i\| \leq \sqrt{2\nu'/\lambda_{\min}(K_i)}$ for all $t \geq T_1$. Based on the error transformation (11), the tracking error e_1 converges to a small neighborhood around zero with guaranteed prescribed performance (3) in a finite time T_1 as well, and all the signals in the closed-loop system are uniformly ultimately bounded.

B. Proof of Theorem 5

Similar to the adaptive neural DSC design in the Section 3, the derivative of z_i ($i = 1, 2, 3, 4$) along (53) and (54) yields

$$\begin{aligned} \dot{z}_1 & = -(C_1 + Y^T Y) z_1 + Y z_2 + Y y_1. \\ \dot{z}_2 & = M^{-1}(q_1) K \left(-Y z_1 - C_2 z_2 + z_3 + y_2 \right. \\ & \quad \left. - \overline{W}_2^T S_2(\Psi_2) + H_2(\Psi_2) - K^{-1} C(q_1, \dot{q}_1) z_2 \right) \\ \dot{z}_3 & = -z_2 - C_3 z_3 + z_4 + y_3 \\ \dot{z}_4 & = J^{-1} \left[-z_3 - C_4 z_4 - \overline{W}_4^T S_4(\Psi_4) + H_4(\Psi_4) \right] \end{aligned} \quad (\text{B.1})$$

According to Theorem 4, there exists a small positive constant ϵ_i' for the recurrent orbit $\psi_{d_i}(x_{d_i}(t))$ ($i = 2, 4$), such

that $\|\overline{W}_i^T S_i(\Psi_i) - H_i(\Psi_i)\| < \epsilon'_i$. Thus, by Young's inequality, the derivative of V along (A.1) and (B.1) yields

$$\begin{aligned} \dot{V} \leq & -\sum_{i=1}^4 c'_i z_i^T K_i z_i - \sum_{l=1}^3 \xi_l y_l^T y_l + (N_1^2 + N_2^2 + N_3^2) \rho \\ & + \frac{\epsilon_2'^2}{4} + \frac{\epsilon_4'^2}{4}, \end{aligned} \quad (\text{B.2})$$

where $c'_1 = \lambda_{\min}(C_1)$, $c'_2 = \lambda_{\min}[(C_2 - 2I)M^{-1}(q_1)K]$, $c'_3 = \lambda_{\min}(C_3 - I)$, $c'_4 = \lambda_{\min}[(C_4 - I)J^{-1}]$. Let $C_2 > 2I$, $C_3 > I$, and $C_4 > I$; (B.2) can be rewritten as

$$\dot{V} \leq -2k'_0 V + b'_0, \quad (\text{B.3})$$

where $b'_0 = (N_1^2 + N_2^2 + N_3^2)\rho + \epsilon_2'^2/4 + \epsilon_4'^2/4$ and $k'_0 = \min\{c'_1, c'_2, c'_3, c'_4, \xi_1, \xi_2, \xi_3\} > 0$. Let $\nu' = b'_0/(2k'_0)$; then, based on the similar analysis in Theorem 3, it can be concluded that the tracking error e_1 converges to a small neighborhood around zero with the prescribed performance, and all the signals in the closed-loop system are uniformly ultimately bounded.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank Professor Chenguang Yang for the experiment platform. Moreover, this work was partially supported by the National Natural Science Foundation of China (nos. 61773169, 61374119, 61473121, and 61611130214), the Royal Society Newton Mobility Grant IE150858, the Guangdong Natural Science Foundation under Grants 2017A030313369 and 2017A030313381, the youth talent of Guangdong Province, and the Fundamental Research Funds for the Central Universities.

References

- [1] M. W. Spong and M. Vidyasagar, *Robot Dynamics and Control*, Chap. 13, Wiley, New York, NY, USA, 1989.
- [2] J. H. Oh and J. S. Lee, "Control of flexible joint robot system by backstepping design approach," in *Proceedings of International Conference on Robotics and Automation*, vol. 4, pp. 3435–3440, 1997.
- [3] J. Huang and J. Lin, "Backstepping Control Design of a Single-Link Flexible Robotic Manipulator," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 11775–11780, 2008.
- [4] Y. Soukkou and S. Labiod, "Adaptive backstepping control using combined direct and indirect adaptation for a single-link flexible-joint robot," *International Journal of Industrial Electronics and Drives*, vol. 2, no. 1, p. 11, 2015.
- [5] J.-J. E. Slotine and S. Hong, "Two-time scale sliding control of manipulators with flexible joints," in *Proceedings of the 1986 American Control Conference*, pp. 805–810, 1986.
- [6] A.-C. Huang and Y.-C. Chen, "Adaptive sliding control for single-link flexible-joint robot with mismatched uncertainties," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 5, pp. 770–775, 2004.
- [7] B. Xu and P. Zhang, "Minimal-learning-parameter Technique Based Adaptive Neural Sliding Mode Control of MEMS Gyroscope," *Complexity*, Article ID 6019175, 8 pages, 2017.
- [8] B. Xu and P. Zhang, "Composite Learning Sliding Mode Control of Flexible-link Manipulator," *Complexity*, Article ID 9430259, 6 pages, 2017.
- [9] C.-I. Morarescu and B. Brogliato, "Passivity-based switching control of flexible-joint complementarity mechanical systems," *Automatica*, vol. 46, no. 1, pp. 160–166, 2010.
- [10] Y. Li, S. Tong, and T. Li, "Adaptive fuzzy output feedback control for a single-link flexible robot manipulator driven DC motor via backstepping," *Nonlinear Analysis: Real World Applications*, vol. 14, no. 1, pp. 483–494, 2013.
- [11] F. Abdollahi, H. A. Talebi, and R. V. Patel, "A stable neural network-based observer with application to flexible-joint manipulators," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 17, no. 1, pp. 118–129, 2006.
- [12] S. J. Yoo, "Distributed adaptive containment control of networked flexible-joint robots using neural networks," *Expert Systems with Applications*, vol. 41, no. 2, pp. 470–477, 2014.
- [13] I. F. Jasim and P. W. Plapper, "Robust direct adaptive fuzzy control of flexible joints robots with time-varying stiffness/damping parameters," in *Proceedings of the Joint Conference of ISR 2014 - 45th International Symposium on Robotics and Robotik 2014 - 8th German Conference on Robotics, ISR/ROBOTIK 2014*, pp. 149–156, June 2014.
- [14] E. Psomopoulou, A. Theodorakopoulos, Z. Doulgeri, and G. A. Rovithakis, "Prescribed Performance Tracking of a Variable Stiffness Actuated Robot," *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1914–1926, 2015.
- [15] L. Zhang, Z. Li, and C. Yang, "Adaptive neural network based variable stiffness control of uncertain robotic systems using disturbance observer," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 3, pp. 2236–2245, 2017.
- [16] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.
- [17] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [18] M. Krstic, I. Kanellakopoulos, and P. Kokotovic, *Nonlinear and Adaptive Control Design*, Wiley, NY, USA, 1995.
- [19] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice-Hall, Upper Saddle River, NJ, USA, 2nd edition, 1999.
- [20] S. S. Ge and C. Wang, "Direct adaptive NN control of a class of nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 13, no. 1, pp. 214–221, 2002.
- [21] D. Swaroop, J. K. Hedrick, P. P. Yip, and J. . Gerdes, "Dynamic surface control for a class of nonlinear systems," *Institute of Electrical and Electronics Engineers Transactions on Automatic Control*, vol. 45, no. 10, pp. 1893–1899, 2000.
- [22] D. Wang and J. Huang, "Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 16, no. 1, pp. 195–202, 2005.
- [23] S. Gao, H. Dong, and B. Ning, "Observer-based nonlinear feedback decentralized neural adaptive dynamic surface control

- for large-scale nonlinear systems," *International Journal of Adaptive Control and Signal Processing*, 2017.
- [24] S. J. Yoo, J. B. Park, and Y. H. Choi, "Adaptive output feedback control of flexible-joint robots using neural networks: Dynamic surface design approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 19, no. 10, pp. 1712–1726, 2008.
- [25] S. Gao, H. Dong, S. Lyu, and B. Ning, "Truncated adaptation design for decentralised neural dynamic surface control of interconnected nonlinear systems under input saturation," *International Journal of Control*, vol. 89, no. 7, pp. 1447–1466, 2016.
- [26] S. Gao, B. Ning, and H. Dong, "Fuzzy dynamic surface control for uncertain nonlinear systems under input saturation via truncated adaptation approach," *Fuzzy Sets and Systems*, vol. 290, pp. 100–117, 2016.
- [27] M. Wang, X. Liu, and P. Shi, "Adaptive neural control of pure-feedback nonlinear time-delay systems via dynamic surface technique," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 41, no. 6, pp. 1681–1692, 2011.
- [28] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.
- [29] W. He, A. O. David, Z. Yin, and C. Sun, "Neural network control of a robotic manipulator with input deadzone and output constraint," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 6, pp. 759–770, 2016.
- [30] C. P. Bechlioulis and G. A. Rovithakis, "Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance," *Institute of Electrical and Electronics Engineers Transactions on Automatic Control*, vol. 53, no. 9, pp. 2090–2099, 2008.
- [31] A. K. Kostarigka, Z. Doulgeri, and G. A. Rovithakis, "Prescribed performance tracking for flexible joint robots with unknown dynamics and variable elasticity," *Automatica*, vol. 49, no. 5, pp. 1137–1147, 2013.
- [32] M. Wang and H. Ye, "Adaptive neural dynamic surface control for flexible joint manipulator with prescribed performance," in *Proceedings of the 2017 29th Chinese Control And Decision Conference (CCDC)*, pp. 5311–5316, Chongqing, China, May 2017.
- [33] M. Wang, C. Wang, P. Shi, and X. Liu, "Dynamic learning from neural control for strict-feedback systems with guaranteed predefined performance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2564–2576, 2016.
- [34] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2017.
- [35] M. Wang, Y. Zhang, and H. Ye, "Dynamic Learning from Adaptive Neural Control of Uncertain Robots with Guaranteed Full-State Tracking Precision," *Complexity*, Article ID 5860649, 14 pages, 2017.
- [36] X. Xia, T. Zhang, Y. Yi, and Q. Shen, "Adaptive prescribed performance control of output feedback systems including input unmodeled dynamics," *Neurocomputing*, vol. 190, pp. 226–236, 2016.
- [37] C. Wang and D. J. Hill, *Deterministic Learning Theory for Identification, Recognition, and Control*, CRC Press, Boca Raton, FL, USA, 2009.
- [38] T. Liu, C. Wang, and D. J. Hill, "Learning from neural control of nonlinear systems in normal form," *Systems & Control Letters*, vol. 58, no. 9, pp. 633–638, 2009.
- [39] S.-L. Dai, C. Wang, and M. Wang, "Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 111–123, 2014.
- [40] M. Wang, C. Wang, and X. Liu, "Dynamic learning from adaptive neural control with predefined performance for a class of nonlinear systems," *Information Sciences*, vol. 279, pp. 874–888, 2014.
- [41] S.-L. Dai, C. Wang, and F. Luo, "Identification and learning control of ocean surface ship using neural networks," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 4, pp. 801–810, 2012.
- [42] M. Wang and A. Yang, "Dynamic learning from adaptive neural control of robot manipulators with prescribed performance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2244–2255, 2017.
- [43] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.
- [44] S.-L. Dai, M. Wang, and C. Wang, "Neural learning control of marine surface vessels with guaranteed transient tracking performance," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 3, pp. 1717–1727, 2016.
- [45] R. M. Sanner and J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 3, no. 6, pp. 837–863, 1992.
- [46] C. Wang and D. J. Hill, "Learning from neural control," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 17, no. 1, pp. 130–146, 2006.
- [47] H. K. Khalil, *Nonlinear Systems*, Prentice-Hall, Englewood Cliffs, NJ, USA, 2002.

Review Article

A Brief Review of Neural Networks Based Learning and Control and Their Applications for Robots

Yiming Jiang,¹ Chenguang Yang,¹ Jing Na,² Guang Li,³ Yanan Li,⁴ and Junpei Zhong⁵

¹Key Laboratory of Autonomous Systems and Networked Control, College of Automation Science and Engineering, South China University of Technology, Guangzhou, China

²Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming, China

³School of Engineering and Material Sciences, Queen Mary University of London, London, UK

⁴School of Engineering and Informatics, University of Sussex, Brighton, UK

⁵National Institute of Advanced Industrial Science and Technology, Tokyo, Japan

Correspondence should be addressed to Chenguang Yang; cyang@ieee.org

Received 15 June 2017; Accepted 1 October 2017; Published 31 October 2017

Academic Editor: Thierry Floquet

Copyright © 2017 Yiming Jiang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

As an imitation of the biological nervous systems, neural networks (NNs), which have been characterized as powerful learning tools, are employed in a wide range of applications, such as control of complex nonlinear systems, optimization, system identification, and patterns recognition. This article aims to bring a brief review of the state-of-the-art NNs for the complex nonlinear systems by summarizing recent progress of NNs in both theory and practical applications. Specifically, this survey also reviews a number of NN based robot control algorithms, including NN based manipulator control, NN based human-robot interaction, and NN based cognitive control.

1. Introduction

In recent years, the research of neural network (NN) has attracted great attention. It is well known that, mammals' brain, which consists of billions of interconnected neurons, has the ability to deal with complex and computationally demanding tasks, such as face recognition, body motion planning, and muscles activities control. Figure 1 shows a cellular structure of a mammalian neuron. Inspired by the neuron structure, artificial NN (ANN) was developed to emulate the learning ability of the biological neurons system [1–3]. The concept of artificial NNs was initially investigated by McCulloch and Pitts in the 1940s [3], where the network is established with a parallel structure. The basically mathematical model of NN consists of three layers, that is, input layer, hidden layer, and output layer, which are of simple parallel computational structure but with appealing learning ability and computational power to predict nonlinear dynamic patterns.

In past decades, the NN technique has been studied extensively in areas such as control engineering, aerospace, medicine, automotive, psychology, economics, energy science, and many other fields [4–7]. It has been reported that NN can approximate any unknown continuous nonlinear function by overlapping the outputs of each neuron. Moreover, the approximation errors could be made arbitrarily small by choosing sufficient neurons. This enables us to deal with control problems for complex nonlinear systems [8–13]. In addition to system modeling and control, NN has also been successfully applied in various fields such as learning [14–17], pattern recognition [18], and signal processing [19]. And NN has been extensively used for functions approximation, such as to compensate for the effect of unknown dynamics in nonlinear systems [20–31]. The NN control has been proved to be effective for controlling uncertain nonlinear systems and demonstrated superiority in many aspects.

Recently, the researchers have focused on the study of robotics for its increasing importance in both industrial

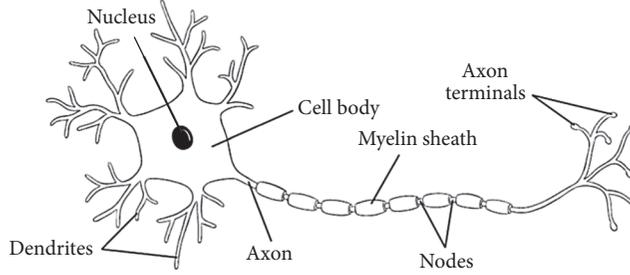


FIGURE 1: An example of mammalian neuron (modified from [32]).

applications and daily life [33–38]. Many advanced robots such as YuMi made by ABB, Baxter made by Rethink, and Rolins’ Justin developed by German Aerospace Agency (DLR) have also been widely allocated. The robots manipulator system is characterized with high-nonlinearity, strong coupling, and time-varying dynamics, thus controlling a robot with not only positioning accuracy, but also enough flexibility to complete a complex task became an interesting yet challenge work. To achieve a high performance control, dynamics of the robot should be known in advance. However, in practice, the robot dynamic model is often rarely known due to the complex robot mechanism, let alone various uncertainties such as parametric uncertainties or modeling errors existing in the robot dynamics. Therefore, advance control algorithm is imperative for next-generation robots. Thanks to the universal approximation and learning ability, the NN has been widely applied in robot control with various applications. The combination of NN and robot controller can provide possible solutions for complex manipulation tasks, for example, robot control with unknown dynamics and robot control with unstructured environment. In this paper, we present a brief review of robot control by means of neural network. The rest of the paper is organized as follows.

After the introduction, in Section 2, we present preliminaries of several popular neural network structures, such as RBFNN and CMAC NN. Section 3 introduces a number of theoretical developments of NN in the fields of adaptive control, optimization, and evolutionary computing. In addition, Section 4 revisits the robot neural network control with the applications in manipulation, human-robot interaction, and robot cognitive control. Section 5 gives a brief discussion about the neural network control and its future research.

2. Preliminaries of Neural Networks

In this section, we will introduce several types of NN structure, which are popularly employed in the control engineering.

2.1. Radial Basis Function Neural Network (RBFNN) [14, 15]. A basic architecture of RBFNN network is shown in Figure 2, which consists of three layers, namely, input layer, hidden layer, and output layer. In the input layer, the NN inputs are applied. In hidden layer, the data is transformed from input space to hidden space, which is always with a higher

dimension. The RBFNN can be used to approximate any continuous vector function, for example, $F(Z)$:

$$\hat{F}(Z) = W^T S(Z), \quad (1)$$

where $\hat{F}(Z)$ is the estimation of $F(Z)$ and Z is NN inputs vector. $\hat{W} = [\hat{W}_1, \hat{W}_2, \dots, \hat{W}_n] \in R^{n \times l}$ is the estimation of NN optimal weight, $S(Z) = [s_1(Z), s_2(Z), \dots, s_l(Z)]^T$ is the regressor, and l denotes the number of NN nodes. Generally, the regressor could be chosen as a Gaussian radical basis function as follows:

$$s_i(\|Z - u_i\|) = \exp \left[\frac{-(Z - u_i)^T (Z - u_i)}{\sigma_i^2} \right], \quad (2)$$

where u_i ($i = 1, \dots, l$) are distinct points in state space and σ_i is the width of Gaussian membership function. It has been well recognized that, using the powerful approximate ability of the RBFNN, we can approximate any continuous nonlinear function over a compact set as

$$F(Z) = W^{*T} S(Z) + \varepsilon, \quad (3)$$

where W^* is the optimal weight vector and ε is the approximate error.

2.2. Cerebellar Model Articulation Controller (CMAC) NN [39]. There has been a predominant tendency to study the learning and control techniques of robots by exploring the principles of biological systems. This is because the biological creatures, mechanisms, and underlying principles are likely to bring novel ideas to improve control performance of the robot in a complex environment. In 1972, Albus proposed a learning mechanism that imitates the structure and function of the cerebellum, called cerebellar model articulation controller (CMAC), which is designed based on a cerebellum neurophysiological model [40]. In comparison to the backpropagation neural network, the CMAC NN was adopted widely in modeling and control of robots system for its rapid learning speed, simple structure, insensitivity of data sequence, and easy implementation [39, 41].

Figure 3 shows the basic structure of the CMAC neural network. The CMAC could be used to approximate the unknown continuous function, $G(Z) = [f_1(Z), f_2(Z), \dots, f_n(Z)]$, where $Z \in R^m$ denotes the m dimensional inputs

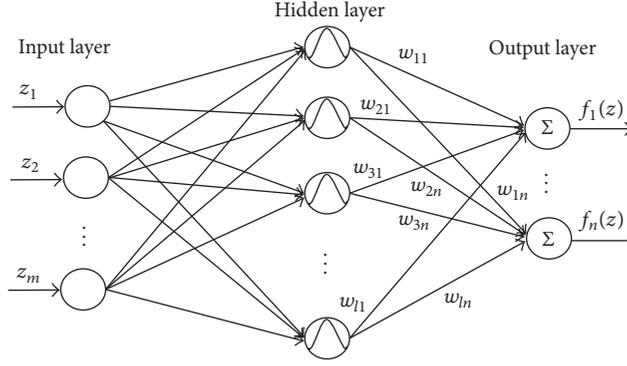


FIGURE 2: Structure of the RBFNN.

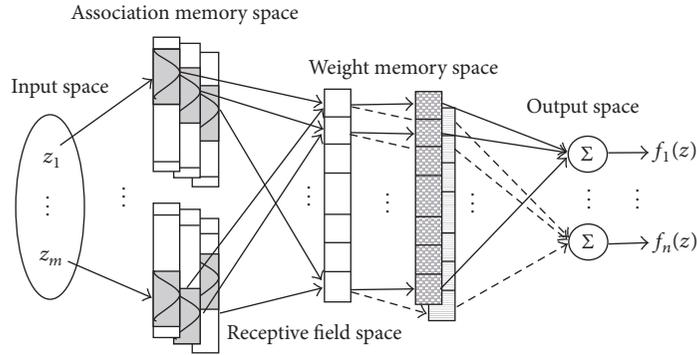


FIGURE 3: Structure of a CMAC neural network.

space. As shown in Figure 3, two components are involved in the CMAC neural network to determine the value of the approximated nonlinear function $G(Z)$:

$$\begin{aligned} R : Z &\longrightarrow C \\ P : C &\longrightarrow F, \end{aligned} \quad (4)$$

where

Z is m -dimensional input space

F is n -dimensional output space

C is N_c -dimensional association space

and $R(\cdot)$ denotes the mapping from the input vector to the association space; that is, $\alpha = R(Z)$. The outputs are computed through $P(\alpha)$, by using a projection of the association vector α onto a weights vector, such that

$$f = P(\alpha) = W^T \alpha, \quad (5)$$

It should be noted that $R(Z)$ can be represented by a multidimensional receptive filed function such that each point in input Z is assigned with an activation value. The

receptive-field basis functions of the association vector could be chosen as Gaussian functions as follows:

$$h_{ik}(\|z_i - u_{ik}\|) = \exp \left[\frac{-(z_i - u_{ik})^2}{\vartheta_{ik}^2} \right], \quad (6)$$

$k = 1, 2, \dots, l,$

where l is number of blocks of the associate space, h_{ik} denotes the k th block associated with the input z_i , u_{ik} denotes the receptive field's center, and ϑ_{ik} is the variance of Gaussian function. Then, the multidimensional receptive-field function can be described as

$$S(Z) = [s_1, s_2, \dots, s_n]^T, \quad (7)$$

where $s_k(Z, u_k, \vartheta_k) = \prod_{i=1}^m h_{ik}(z_i)$, $u_k = [u_{1k}, u_{2k}, \dots, u_{mk}]^T$, and $\vartheta_k = [\vartheta_{1k}, \vartheta_{2k}, \dots, \vartheta_{mk}]^T$. The following property shows the approximation ability provided by the CMAC neural network.

Lemma 1. For a continuous nonlinear function $F(Z)$, there exists an ideal weight value W^* , such that $F(Z)$ could be uniformly approximated by a CMAC with the multiplication of the optimal weights W^* and the associate vector $S(Z)$ as

$$F(Z) = W^T S(Z) + \varepsilon, \quad (8)$$

where ε is the NN construction errors and satisfied $\|\varepsilon\| \leq \varepsilon_N$ and ε_N is a small bounded positive value.

3. Theoretical Developments

3.1. Adaptive Neural Control. During the past two decades, various neural networks have been incorporated into adaptive control for nonlinear systems with unknown dynamics. In [42], a multiplayer discrete-time neural network controller was constructed for a class of multi-input multioutput (MIMO) dynamical systems, where NN weights were trained using an improved online tuning algorithm. An adaptive NN output feedback control was proposed to control two classes of discrete-time systems in the presence of unknown control directions [4]. A robust adaptive neural controller was developed for a class of strict-feedback systems in [43], where a Nussbaum gain technique was employed to deal with unknown virtual control coefficients. A dynamic recurrent NN was employed for construction of an adaptive observer with online turned weights parameters in [44] and to deal with the time-delay of a class of nonlinear dynamical systems in [45]. The time-delay of strict-feedback nonlinear systems was also addressed by using NN control with proper designed Lyapunov-Krasovskii functions in [46]. For a class of unknown nonlinear affine time-delay systems, an adaptive control scheme was proposed by constructing two high-order NNs for identifying system uncertainties [47]. This idea has been further extended to affine nonlinear systems with input time-delay in [48].

It should be noticed that, piecewise continuous functions such as frictions, backlash, and dead-zone are widely existed in industrial plants. Other than continuous nonlinear function, the approximation of these piecewise functions is more challenging since the NN's universal approximation only holds for continuous functions. To approximate these piecewise continuous functions, a novel NN structure was designed by involving a standard activation function and a jump approximation basis function [49]. In [47], a CMAC NN was employed for the closed-loop control of nonlinear dynamical systems with rigorous stability analysis, and in [50] a robust adaptive neural network control scheme was developed for cooperative tracking control of higher-order nonlinear systems.

The adaptive NN control scheme was also proposed for pure-feedback systems. In [51], a high-order sliding mode observer was proposed to estimate the unknown system states while two NNs were constructed to deal with approximation errors and the unknown nonlinearities, respectively. In comparison to the conventional control design for pure-feedback systems, the state-feedback control was achieved without using the backstepping technique. In [52], a neural control framework was proposed for nonlinear servo mechanism to guarantee both the steady-state and transient tracking performance. In this work, a prescribed performance function was employed in an output error transformation, such that the tracking performance can be guaranteed by the regulation control of the outputs. In [53], an adaptive neural control was also designed for a class of nonlinear systems in the presence of time-delays and input dead-zone, and high-order neural networks were employed to deal the unknown uncertainties. In this work, a salient feature lies in the fact that only the norm of the NNs' weights (a scalar) needs to be online

updated, such that the computational efficiency in the online implementation could be significantly improved. In [54], the authors developed a neural network based feedforward control to compensate for the nonlinearities and uncertainties of a dynamically substructured system consisting of both numerical and physical substructures, where an adaptive law with a new leakage term of NN weights error information was developed to achieve improved convergence. An experiment on a quasi-motorcycle testing rig validated the efficacy of this control strategy. In [55], a neural dynamic control was incorporated into the strict-feedback control of a class of unknown nonlinear systems by using the dynamic surface control technique. For a class of uncertain nonlinear systems with unknown hysteresis, NN was used for compensation of the nonlinearities [56]. In [57], to deal with unknown nonsymmetrical input saturations of unknown nonaffine systems, NNs were used in the state/output feedback control based on the mean value theorem and the implicit function. To avoid using the backstepping synthesis, a dynamic surface control scheme was designed by combining the NN with a nonlinear disturbance observer [58].

3.2. NN Based Adaptive Dynamic Programming. In addition to adaptive control, neural networks have also been adopted to solve the optimization problem for nonlinear systems. In convention optimal control, the dynamic programming method was widely used. It aims to minimize a predefined cost function, such that a sequence of optimal control inputs could be derived. However, the cost function is usually difficult to online calculate due to the computation complexity in obtaining the solution of the Hamilton-Jacobi-Bellman (HJB) equation. Therefore, an adaptive/approximate dynamic programming (ADP) technique was developed in [59], where a NN was trained to estimate the cost function and then to derive solutions for the ADP. Generally, the ADP has several different synonyms, including approximate dynamic programming, heuristic dynamic programming (HDP), critic network, and reinforcement learning (RL) [60–62]. Figure 4 shows the basic framework of the HDP with a critic-actor structure. In [63], a discrete-time HJB equation was solved using an NN based HDP algorithm to derive the optimal control of nonlinear discrete-time systems. In [64], three neural networks were constructed for an iterative ADP, such that optimal feedback control of a discrete-time affine nonlinear system could be realized. In [65], a globalized dual heuristic programming was presented to address the optimal control of discrete-time systems. In each iteration, three neural networks were used to learn the cost function and the unknown nonlinear systems. In [66], a reference network combining with an action network and a critic network was introduced in the ADP architecture to derive an internal goal representation, such that the learning and optimization process could be facilitated. The reference network has also been introduced in the online action-dependent heuristic dynamic programming by employing a dual critic network framework. A policy iteration algorithm was introduced for infinite horizon optimal control of nonlinear systems using ADP in [67]. In [68], a reinforcement learning method was introduced for the stabilizing control of uncertain nonlinear

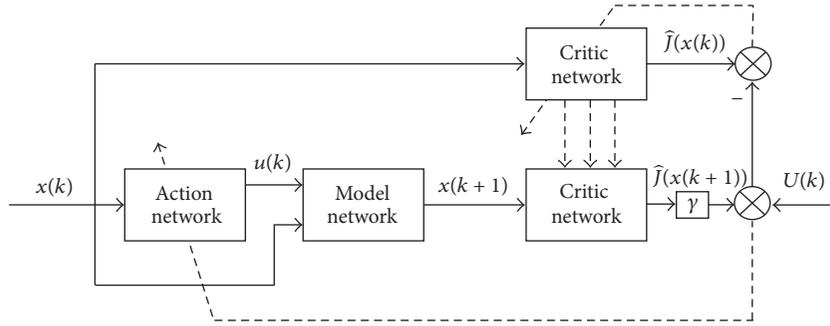


FIGURE 4: An overview of the HDP structure.

systems in the presence of input constraints. By using this RL-based controller, a constrained optimal control problem was solved with construction of only one critic neural network. In [69], an ADP technique for online control and learning of a generalized multiple-input-multiple-output (MIMO) system was investigated. In [70], an adaptive NN based ADP control scheme was presented for a class of nonlinear systems with unknown dynamics. The optimal control law was calculated by using a dual neural network scheme with a critic NN and an identifier NN. Particularly, parameters estimation error was used to online identify the learning weights to achieve the finite-time convergence. Optimal tracking control for a class of nonlinear systems was investigated in [71], where a new “identifier-critic” based ADP framework was proposed.

3.3. Evolutionary Computing. In addition to the capacity of approximation and optimization of the NN, there has been also a great interest in using the evolutionary approaches to train the neural networks. With the evolution of NN architectures, learning rules, connection weights, and input features, an evolutionary artificial neural network (EANN) was designed to provide superior performance in comparison to conventional training approaches [72]. A literature review of the EANN was given in [73], where the evolution strategies such as feedforward artificial NN and genetic algorithms (GA) have been introduced for the EANNs. In [72], several EANN frameworks were introduced by embedding the evolution algorithms (EA) to evolve the NN structure. In [74], an EPNet evolution system was proposed for evolving the feedforward NN based on Fogel’s evolutionary programming (EP) method, which could improve the NN’s connection weights and architectures at the same time as well as decrease noise in the fitness evaluation. Good generalization ability of the evolved NN has been constructed and verified in the experiments. In [75], a GA based technique has been employed to train the NNs in direct neural control systems such that the NN architectures could be optimized. A deficiency of the EANN is that the optimization process would often result in a low training speed. To overcome this problem and facilitate adaptation processes, a hybrid multiobjective evolutionary method was developed in [76], where the singular-value-decomposition (SVD) technique was employed to choose the necessary neurons number in the training of a feedforward NN. The evolutionary approach was

applied to identify a grey-box model with a multiobjective optimization between the clearly known practical systems and approximated nonlinear systems [77]. Applications of evolutionary algorithms for robotic navigation have been introduced and investigated in [78]. A survey of machine learning technique was reported in [79], where several methods to improve the evolutionary computation were reviewed.

The evolution algorithms have been employed in many aspects for evolutions of NNs, such as to train the NN connection weights or to obtain near-optimal NN architectures, as well as adapting learning rules of NNs to their environment. In a word, the evolution algorithms provide NNs with the ability of learning to learn and also to build the relationship between evolution and learning, such that the EANN could perform favorable ability to adapt to changes of the dynamic environment.

4. Applications in Robots

4.1. NN Based Robotic Manipulator Control. Generally speaking, the control methods for robot manipulators can be roughly divided into two groups, model-free control and model based control. For the model-free control approaches like proportional-integral-derivative (PID) control, satisfactory control performance may not be guaranteed. In contrast, the model based control approaches exhibit better control behavior but heavily depend on the validity of the robot model. In practice, however, a perfect robotic dynamic model is always not available due to the complex mechanisms and uncertainties. Additionally, the payload may be varied according to different tasks, which makes the accurate dynamics model hard to be obtained in advance. To solve such problems, the NN approximation-based control methods have been used extensively in applications of robot manipulator control. A basic structure of the adaptive neural network control for robot manipulator is shown in Figure 5. Consider a dynamic model of a robot manipulator given as follows [80]:

$$M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + G(q) = \tau, \quad (9)$$

where $M(q)$, $C(\dot{q}, q)$, and $G(q)$ are the inertial matrix, Coriolis matrix, and gravity vector, respectively. Then NN control design could be given as follows:

$$\tau_d = -K_1 e_1 - K_2 e_2 + \widehat{W}S(Z), \quad (10)$$

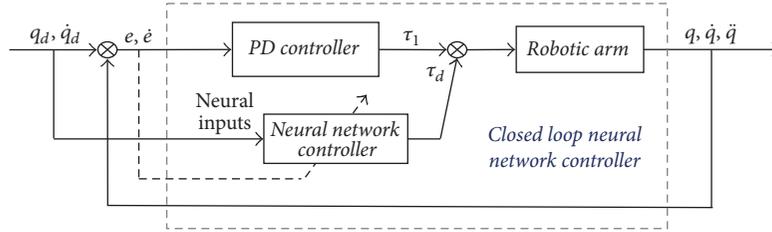


FIGURE 5: A basic structure of the adaptive NN robot control.

where e_1 is the tracking error, e_2 is the velocity tracking error, $\widehat{W}S(Z)$ is the NN controller with \widehat{W} being the weights matrix and $S(Z)$ being the NN regressor vector, and K_1 and K_2 are control gains specified by the designer.

From (10), we can see that the robot controller consists of a PD-like controller and a NN controller. In traditional model based controllers, the dynamic model of the robot could be regarded as a feedforward to address the effect caused by the robot motion. In practice, however, $M(q)$, $C(q, \dot{q})$, and $G(q)$ may not be known. Therefore, the NNs are used to approximate the unknown dynamics $f = M(q)\ddot{q} + C(\dot{q}, q)\dot{q} + G(q)$ and to improve the performance of the system via the online estimation. To adapt the NN weights, adaptive laws are designed as follows:

$$\dot{\widehat{W}} = \Gamma (Se - \sigma \widehat{W}), \quad (11)$$

where Γ and σ are specified positive parameters. The last term of right-hand side of (11) is the sigma modification, which is used to enhance the convergence and robustness of the parameters adaptation.

In [80], a NN based share control method was developed to control a teleoperated robot with environmental uncertainties. In this work, the RBFNN was constructed to compensate for the unknown dynamics of the teleoperated robot. Particularly, a shared control strategy was developed into the controller to achieve the automatic obstacle avoidance combining with the information of visual camera and the robot body, such that the obstacle could be successfully avoided and the operator could focus more on the operated task rather than the environment to guarantee the stability and manipulation. In addition, error transformations were integrated into the adaptive NN control to guarantee the transient control performance. It was shown that, by using the NN technique, the control performance in both kinematic level and dynamic level of the teleoperated robot was enhanced. In [81], an extreme learning machine (ELM) based control strategy was proposed for uncertain robot manipulators to identify both the elasticity and geometry of an object. This ELM was applied to deal with the unknown nonlinearity of the robot manipulator to enhance the control performance. Particularly, by utilizing ELM, the proposed controller could guarantee that the robot dynamics follow a reference model, such that the desired set point and the feedforward force could be updated to estimate the geometry and stiffness of the object. As a result, the reference model could be exactly matched with a limited number of iterations.

In [82], the NN controller was also employed to control a wheel inverted pendulum, which has been decomposed into two subsystems, a fully actuated second-order planar moving subsystem and a passive first-order pendulum subsystem. Then the RBFNN was employed to compensate for the uncertain dynamics of the two subsystems by using its powerful learning ability, such that the enhanced control performance could be realized by using the NN learning. In [83], a global adaptive neural control was proposed for a class of robot manipulators with finite-time convergence learning performance. This control scheme employed a smooth switching mechanism combining with a nominal neural network controller and a robust controller to ensure global uniform ultimately bounded stability. The optimal weights were obtained by the finite-time estimation algorithm such that, after the learning process, the learning weights could be reused next time for repeated tasks. The global NN control mechanism has been further extended to the control of dual arm robot manipulator in [84], where knowledge of both robot manipulator and the grasping object is unavailable in advance. By integrating prescribed functions into the design of controller, the transient performance of the dual arm robot control was regularly guaranteed. The NN was also employed to deal with synchronization problem of multiple robot manipulators in [85], where the reference trajectories are only available for part of the team members. By using the NN approximation controller, the robot has shown better control performance with enhanced transient performance and enhanced robustness. A RBFNN was constructed to compensate for the nonlinear terms of a five-bar manipulator based on an error transformation function [86]. The NN control was also applied in the robot teleoperation control [87, 88]. Moreover, a NN approximation technique was employed to deal with the unknown dynamics, kinematics, and actuator properties in the manipulator tracking control [89].

4.2. NN Based Robot Control with Input Nonlinearities. Another challenge of the robot manipulator is that the input nonlinearities such as friction, dead-zone, and actuator saturation may inevitably exist in the robot systems. These input nonlinearities may lead to larger tracking errors and degeneration of the control performance. Therefore, a number of works have been proposed to handle the nonlinearities by utilizing the neural network design. A neural adaptive

controller was designed to deal with the effect of input saturation of the robot manipulator in [90] as follows:

$$\begin{aligned} \tau = & -z_1 + \widehat{W}_D S_D(Z_D) \alpha_1 + \widehat{W}_C S_C(Z_C) \alpha_1 \\ & + \widehat{W}_G S_G(Z_G) + K_p(z_2 + \xi) + K_r \operatorname{sgn}(z_2), \end{aligned} \quad (12)$$

where z_1 is the robot position tracking error, z_2 is the velocity tracking error, and α_1 is an auxiliary controller. \widehat{W}_D , \widehat{W}_C , and \widehat{W}_G are the NN weights, $S_D(Z_D)$, $S_C(Z_C)$, and $S_G(Z_G)$ are the NN regressor vectors, and K_p and K_r are control gains specified by the designer. ξ is an auxiliary system designed to reduce the effect of the saturation with $\dot{\xi}$ defined as follows.

$$\begin{aligned} \dot{\xi} = & \begin{cases} -K_\xi \xi - \frac{|z_2^T \Delta \tau| + (1/2) \Delta \tau^T \Delta \tau}{\|\xi\|^2} \xi + \Delta \tau & \|\xi\| \geq \mu \\ 0 & \|\xi\| < \mu \end{cases} \end{aligned} \quad (13)$$

where $\Delta \tau$ is the torque error caused by saturation, and K_ξ is a small positive value. To update the NN weights, adaptive laws are designed as follows:

$$\begin{aligned} \dot{\widehat{W}}_D &= \Gamma_{Dk} (S_{Dk} \alpha_1 e_{2k} - \sigma_{Dk} \widehat{W}_{Dk}) \\ \dot{\widehat{W}}_C &= \Gamma_{Ck} (S_{Ck} \alpha_1 e_{2k} - \sigma_{Ck} \widehat{W}_{Ck}) \\ \dot{\widehat{W}}_G &= \Gamma_{Gk} (S_{Gk} \alpha_1 e_{2k} - \sigma_{Gk} \widehat{W}_{Gk}), \end{aligned} \quad (14)$$

where Γ_{Dk} , Γ_{Ck} , and Γ_{Gk} are specified positive parameters and σ_{Dk} , σ_{Ck} , and σ_{Gk} are positive parameters.

In [91], an adaptive neural network controller was constructed to approximate the input dead-zone and the uncertain dynamics of the robotic manipulator, while the output constraint was also considered in the feedback control. In [92], the NN was applied for the estimation of the unknown model parameters of a marine surface vessel and in [93] the full-state constraint of an n -link robotic manipulator was achieved by using the NN control. The NN controller was also constructed for flexible robotic manipulators to deal with the vibration suppression based on a lumped spring-mass model [94] while in [95], two RBFNNs were constructed for flexible robot manipulators to compensate for the unknown dynamics and the dead-zone effect, respectively.

The NN has also been used in many important industrial fields, such as autonomous underwater vehicles (AUVs) and hypersonic flight vehicle (HFV). In [96], the NN has been constructed to deal with the attitude of AUVs in the presence of input dead-zone and uncertain model parameters. In [97], the adaptive neural control was employed to deal with underwater vehicle control in discrete-time domain encountered with the unknown input nonlinearities, external disturbance, and model uncertainties. Then the reinforcement learning was applied to address these uncertainties by using a critic NN and an action NN. The hypersonic flight vehicle control was investigated in [98] where the aerodynamic uncertainties and unknown disturbances were addressed by a disturbance

observer based NN. In [99], a neural learning control was embedded in the HFV controller to achieve the global stability via a switching mechanism and a robust controller.

4.3. NN Based Human-Robot Interaction Control. Recently, there is a predominant tendency to employ the robots in the human-surrounded environment, such as household services or industrial applications, where humans and robots may interact with each other directly. Therefore, interaction control has become a promising research field and has been widely studied. In [100], a learning method was developed such that the dynamics of a robot arm could follow a target impedance model with only knowledge of the robotic structure (see Figure 6).

The NN was further employed in robot control in interaction with an environment [101], where impedance control was achieved with the completely unknown robotic dynamics. In [102], a learning method was developed such that the robot was able to adjust the impedance parameters when it interacted with unknown environments. In order to learn optimal impedance parameters in the robot manipulator control, an adaptive dynamic programming (ADP) method was employed when the robot interacted with unknown time-varying environments, where NNs were used for both critic and actor networks [103]. The ADP was also employed for coordination of multirobots [104], in which possible disagreement between different manipulators was handled and dynamics of both robots and the manipulated object were not required to be known.

In this work, the controller consists of two parts, a critic network which was used to approximate the cost function, and an actual NN which was designed to control the robot. The critic NN is designed as follows [104]:

$$\widehat{Y}(t) = \widehat{W}_C S_C(Z_C), \quad (15)$$

where $Z_C = \xi$, $\xi = [\dot{x}_o^T, z^T, x_o^T]^T$ with x_o being the position of the object and z being the tracking error, \widehat{W}_C is the NN weight, and S_C is the regressor vector.

The critic NN is used to approximate a cost function $c(t) = \xi^T Q \xi + u^T R u$, where u denotes the control input, and Q and R are positive definite matrix. Since the control objective is to minimize the control effort, the adaptation law is designed as

$$\dot{\widehat{W}}_c = -\sigma_c \frac{\partial E_c}{\partial \widehat{W}_c} = \sigma_c (c - \widehat{W}_c^T \dot{S}_c) \dot{S}_c, \quad (16)$$

where σ_c is the learning rate and $E_c = (1/2)(c - \widehat{W}_c^T \dot{S}_c)^2$.

On the other hand, the actual NN control is designed to control the robot as

$$u = \widehat{W}_a^T S_a(Z_a) - e - K_2 e_v, \quad (17)$$

where $\widehat{W}_a^T S_a(Z_a)$ could learn the dynamics of the robot, with \widehat{W}_a being the NN weight and S_a being the regressor vector. e and e_v are the position and velocity tracking errors, respectively, and K_2 is the control gain.

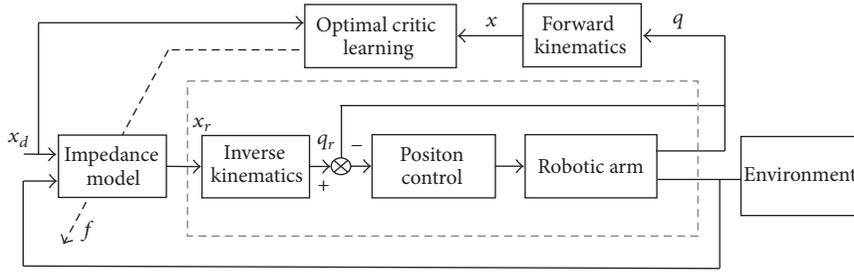


FIGURE 6: Block of the learning impedance control.

Since the control objective is to guarantee the estimation of both robot dynamics and the cost function $\hat{Y}(t)$, the adaptive law is selected as follows:

$$\dot{\hat{W}}_{a,i}^T = -\sigma_a (\hat{W}_{a,i}^T S_a + k_r \hat{Y}) S_a, \quad (18)$$

where σ_a and k_r are positive constants.

On the other hand, as a fundamental element of the next-generation robots, the human-robot collaboration (HRC) has been widely studied by roboticists and NN is employed in HRC with its powerful learning ability. In [105], the NNs were employed to estimate the human partner's motion intention in human-robot collaboration, such that the robot was able to actively follow its human partner. To adjust the robot's role to lead or to follow according to the human's intention, game theory was employed for fundamental analysis of human-robot interaction and an adaptation law was developed in [106]. Policy iteration combining with NN was adopted to provide a rigorous solution to the problem of the system equilibrium in human-robot interaction [107].

4.4. NN Based Robot Cognitive Control. According to the predictive processing theory [108], the human brain is always actively anticipating the incoming sensorimotor information. This process exists because the living beings exhibit latencies due to neural processing delays and a limited bandwidth in their sensorimotor processing. To compensate for such a delay, in human brain, neural feedback signals (including lateral and top-down connections) modulate the neural activities via inhibitory or excitatory connections by influencing the neuronal population coding of the bottom-up sensory-driven signals in the perception-action system. Similarly, in robotic systems, it is claimed that such a delay and a limited bandwidth also can be compensated by the predictive functions learnt by recurrent neural models. Such a learning process can be done via only visual processing [109] or in the loop of perception and action [110].

Based on the hierarchical sensorimotor integration theory, which advocates that action and perception are intertwined by sharing the same representational basis [111], the representation on different levels of sensory perception does not explicitly represent actions; instead, there is an encoding of the possible future percept which is learnt from prior sensorimotor knowledge.

In the Bayesian, once this perception and action links have been established after learning, these perception-action

associations in this architecture allow the following operations.

First, these associations allow predicting the perceptual outcome of given actions by means of the forward models (e.g., Bayesian model). It can be written as

$$P(E | A, I) \propto P(A | E) P(E | I), \quad (19)$$

where E estimates the upcoming perception evidence given an executed action A and other prior information you have already known in I . The term $P(E | A, I)$ suggests that a prelearnt model representing the possibility of a motor action A will be executed given that a (possible) resulting sensory evidence E is perceived (backward computation).

Second, these associations allow selecting an appropriate movement given an intended perceptual representation. From the backward computations introduced in the following equation, a predictive sensorimotor integration occurs:

$$P(A | E, G) \propto P(E | A) P(A | G), \quad (20)$$

where A indicates a particular action selected given the (intended) sensory information E and a goal G . Here we assume that one's action is only determined by the current sensory input and the goal.

In terms of its hierarchical organization, it also allows this operation: with bidirectional information pathways, a low level perception representation can be expressed on a higher level, with a more complex receptive field, and vice versa $e_{\text{low}} \leftrightarrow e_{\text{high}}$. This can be realized by the bidirectional deep architectures such as [112]. Conceptually, these operations can be achieved by extracting statistical regularity shown in Figure 7.

Since both perception and action processes can be seen as temporal sequences, from the mathematical perspective, the recurrent networks are Turing-Complete and have a learning capacity to learn time sequences with arbitrary length [113], if properly trained. Furthermore, such recurrent connections can be placed in a hierarchical way in which the prediction functions on different layers attempt to predict the nonlinear time-series in different time-scales [114]. From this point, the recurrent neural network with parametric bias units (RNNPB) [115] and multiple time-scale recurrent neural networks (MTRNN) [116] were applied to predict sequences by understanding them in various temporal levels.

The difference of the temporal levels controls the properties of the different levels of the presentation in the deep

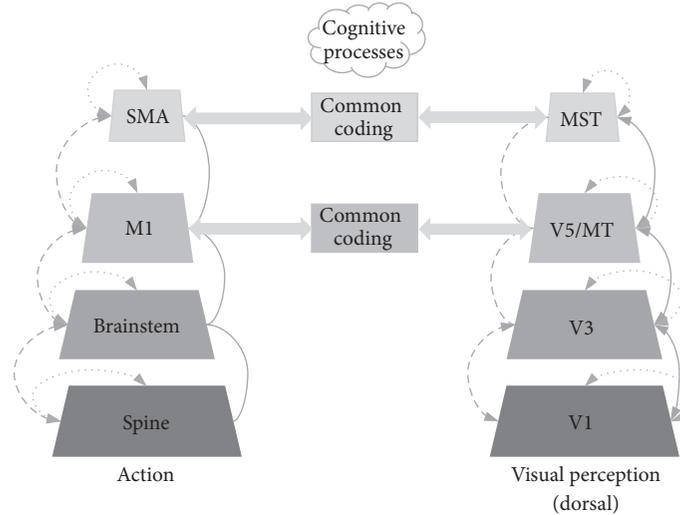


FIGURE 7: The process of cognitive control.

recurrent network. For instance, in the MTRNN network [112], the learning of each neuron follows the updating rule of classical firing rate models, in which the activity of a neuron is determined by the average firing rate of all the connected neurons. Additionally, the neuronal activity is also decaying over time following an updating rule of leaky integrator model. Assuming the i -th MTRNN neuron has the number of N connections, the current membrane potential status of a neuron can be defined by both the previous activation and the current synaptic inputs:

$$u_{i,t+1} = \left(1 - \frac{1}{\tau_i}\right) u_{i,t} + \frac{1}{\tau_i} \left[\sum w_{i,j} x_{j,t}\right], \quad (21)$$

where $w_{i,j}$ represents the synaptic weight from the j -th neuron to the i -th neuron, $x_{j,t}$ is the activity of j -th neuron at t -th time-step, and τ is the time-scale parameter which determines the decay rate of this neuron: a larger τ means their activities change slowly over time compared with those with a smaller time-scale parameter τ .

In [117], the concepts of predictive coding were discussed in detail, where the learning, generation, and recognition of actions can be conducted by means of the principle of prediction error minimization. By using the predictive coding, the RNNPB and MTRNN are capable for both generating own actions and recognizing the same actions performed by others. Recently, the study on neurorobotics experiments has shown that the dynamic predictive coding scheme can be used to address fluctuations in temporal patterns when training a recurrent neural network (RNN) model [118]. This predictive coding scheme enables organisms to predict perceptual outcomes based on current intentions of actions to the external environment and to forecast perceptual sequences corresponding to given intention states [118].

Based on this architecture, two-layer RNN models were utilized to extract visual information [119] and to understand intentions [120] or emotion status [121] in social robotics; three-layer RNN models were used to integrate and

understand multimodal information for a humanoid iCub robot [112, 122]. Moreover, the predictive coding framework has been extended to variational Bayes predictive coding MTRNN, which can arbitrate between deterministic model and probabilistic model by setting a metaparameter [123]. Such extension could provide significant improvement in dealing with noisy fluctuated sensory inputs which robots are expected to experience in more real world setting. In [124], a MTRNN was employed to control a humanoid robot and experimental results have shown that, by using only partial training data, the control model can achieve generalization by learning in a lower feature perception level.

The hierarchical structure of RNN exhibits a great learning capacity to store multimodal information which is beneficial for the robotic systems to understand and to predict in a complex environment. As the future models and applications, the state-of-the-art deep learning techniques or the motor actions of robotic systems can be further integrated into this predictive architecture.

5. Conclusion

In summary, great achievements for control design of nonlinear system by means of neural networks have been gained in the last two decades. Despite the impossibility in identifying or listing all the related contributions in this short review, efforts have been made to summarize the recent progress in the area of NN control and its particular applications in the robot learning control, the robot interaction control, and the robot recognition control. In this paper, we have shown that significant progress of NN has been made in control of the nonlinear systems, in solving the optimization problem, in approximating the system dynamics, in dealing with the input nonlinearities, in human-robot interaction, and in the pattern recognition. All these developments accompany not only the development of techniques in control and advanced manufactures, but also theatrical progress in constructing and developing the neural networks. Although huge efforts

have been made to embed the NN in practical control systems, there is still a large gap between the theory and practice. To improve the feasibility and usability, the evolutionary computing theory has been proposed to train the NNs. It can automatically find a near-optimal NN architecture and allow a NN to adapt its learning rule to its environment. However, the complex and long training process of the evolutionary algorithms deters their practical applications. More efforts need to be made to evolve the NN architecture and NN learning technique in the control design. On the other hand, in human brain, the neural activities are modulated via inhibitory or excitatory connections by influencing the neuronal population coding of the bottom-up sensory-driven signals in the perception-action system. In this sense, how to integrate the sensor-motor information into the network to make NNs more feasible to adapt to the environment and to resemble the capacity of the human brain deserves further investigations.

In conclusion, a brief review on neural networks for the complex nonlinear systems is provided with adaptive neural control, NN based dynamic programming, evolution computing, and their practical applications in the robotic fields. We believe this area may promote increasing investigations in both theories and applications. And emerging topics, like deep learning [125–128], big data [129–131], and cloud computing, may be incorporated into the neural network control for complex systems; for example, deep neural networks could be used to process massive amounts of unsupervised data in complex scenarios, neural networks can be helpful in reducing the data dimensionality, and the optimization of NN training may be employed to enhance the learning and adaptation performance of robots.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Nature Science Foundation (NSFC) under Grant 61473120, Guangdong Provincial Natural Science Foundation, 2014A030313266, International Science and Technology Collaboration, Grant 2015A050502017, Science and Technology Planning Project of Guangzhou, 201607010006, State Key Laboratory of Robotics and System (HIT) Grant SKLRS-2017-KF-13, and the Fundamental Research Funds for the Central Universities.

References

- [1] F. Gruau, D. Whitley, and L. Pyeatt, "A comparison between cellular encoding and direct encoding for genetic neural networks," in *Proceedings of the 1st annual conference on genetic programming*, pp. 81–89, 1996.
- [2] H. de Garis, "An artificial brain ATR's CAM-Brain Project aims to build/evolve an artificial brain with a million neural net modules inside a trillion cell Cellular Automata Machine," *New Generation Computing*, vol. 12, no. 2, pp. 215–221, 1994.
- [3] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biology*, vol. 5, no. 4, pp. 115–133, 1943.
- [4] C. Yang, S. S. Ge, C. Xiang, T. Chai, and T. H. Lee, "Output feedback NN control for two classes of discrete-time systems with unknown control directions in a unified approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 19, no. 11, pp. 1873–1886, 2008.
- [5] R. Kumar, R. K. Aggarwal, and J. D. Sharma, "Energy analysis of a building using artificial neural network: A review," *Energy and Buildings*, vol. 65, pp. 352–358, 2013.
- [6] C. Alippi, C. De Russis, and V. Piuri, "A neural-network based control solution to air-fuel ratio control for automotive fuel-injection systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 33, no. 2, pp. 259–268, 2003.
- [7] E. M. Azoff, *Neural network time series forecasting of financial markets*, John Wiley & Sons, Inc, 1994.
- [8] I. Fister, P. N. Suganthan, J. Fister et al., "Artificial neural network regression as a local search heuristic for ensemble strategies in differential evolution," *Nonlinear Dynamics*, vol. 84, no. 2, pp. 895–914, 2016.
- [9] Y. Zhang, S. Li, and H. Guo, "A type of biased consensus-based distributed neural network for path planning," *Nonlinear Dynamics*, vol. 89, no. 3, pp. 1803–1815, 2017.
- [10] X. Shi, Z. Wang, and L. Han, "Finite-time stochastic synchronization of time-delay neural networks with noise disturbance," *Nonlinear Dynamics*, vol. 88, no. 4, pp. 2747–2755, 2017.
- [11] P. He and Y. Li, "H ∞ synchronization of coupled reaction-diffusion neural networks with mixed delays," *Complexity*, vol. 21, no. S2, pp. 42–53, 2016.
- [12] Z. Tu, J. Cao, A. Alsaedi, F. E. Alsaadi, and T. Hayat, "Global Lagrange stability of complex-valued neural networks of neutral type with time-varying delays," *Complexity*, vol. 21, no. S2, pp. 438–450, 2016.
- [13] C. Wang, S. Guo, and Y. Xu, "Formation of autapse connected to neuron and its biological function," *Complexity*, vol. 2017, Article ID 5436737, 9 pages, 2017.
- [14] J. D. J. Rubio, I. Elias, D. R. Cruz, and J. Pacheco, "Uniform stable radial basis function neural network for the prediction in two mechatronic processes," *Neurocomputing*, vol. 227, pp. 122–130, 2017.
- [15] J. d. Rubio, "USNFIS: Uniform stable neuro fuzzy inference system," *Neurocomputing*, vol. 262, pp. 57–66, 2017.
- [16] Q. Liu, J. Yin, V. C. M. Leung, J.-H. Zhai, Z. Cai, and J. Lin, "Applying a new localized generalization error model to design neural networks trained with extreme learning machine," *Neural Computing and Applications*, pp. 1–8, 2014.
- [17] R. J. de Jesús, "Interpolation neural network model of a manufactured wind turbine," *Neural Computing and Applications*, pp. 1–12, 2016.
- [18] C. Mu and D. Wang, "Neural-network-based adaptive guaranteed cost control of nonlinear dynamical systems with matched uncertainties," *Neurocomputing*, vol. 245, pp. 46–54, 2017.
- [19] Z. Lin, D. Ma, J. Meng, and L. Chen, "Relative ordering learning in spiking neural network for pattern recognition," *Neurocomputing*, 2017.
- [20] J. Yu, J. Sang, and X. Gao, "Machine learning and signal processing for big multimedia analysis," *Neurocomputing*, vol. 257, pp. 1–4, 2017.

- [21] T. Zhang, S. S. Ge, and C. C. Hang, "Adaptive neural network control for strict-feedback nonlinear systems using backstepping design," *Automatica*, vol. 36, no. 12, pp. 1835–1846, 2000.
- [22] S. S. Ge and T. Zhang, "Neural-network control of nonaffine nonlinear system with zero dynamics by state and output feedback," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 14, no. 4, pp. 900–918, 2003.
- [23] S. S. Ge, C. Yang, and T. H. Lee, "Adaptive predictive control using neural network for a class of pure-feedback systems in discrete time," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 19, no. 9, pp. 1599–1614, 2008.
- [24] F. W. Lewis, S. Jagannathan, and A. Yesildirak, *Neural network control of robot manipulators and non-linear systems*, CRC Press, 1998.
- [25] S. Jagannathan and F. L. Lewis, "Identification of nonlinear dynamical systems using multilayered neural networks," *Automatica*, vol. 32, no. 12, pp. 1707–1712, 1996.
- [26] D. Vrabie and F. Lewis, "Neural network approach to continuous-time direct adaptive optimal control for partially unknown nonlinear systems," *Neural Networks*, vol. 22, no. 3, pp. 237–246, 2009.
- [27] C. Yang, S. S. Ge, and T. H. Lee, "Output feedback adaptive control of a class of nonlinear discrete-time systems with unknown control directions," *Automatica*, vol. 45, no. 1, pp. 270–276, 2009.
- [28] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [29] Y. Jiang, C. Yang, and H. Ma, "A review of fuzzy logic and neural network based intelligent control design for discrete-time systems," *Discrete Dynamics in Nature and Society*, Article ID 7217364, Art. ID 7217364, 11 pages, 2016.
- [30] Y. Jiang, C. Yang, S.-L. Dai, and B. Ren, "Deterministic learning enhanced neural network control of unmanned helicopter," *International Journal of Advanced Robotic Systems*, vol. 13, no. 6, pp. 1–12, 2016.
- [31] Y. Jiang, Z. Liu, C. Chen, and Y. Zhang, "Adaptive robust fuzzy control for dual arm robot with unknown input deadzone nonlinearity," *Nonlinear Dynamics*, vol. 81, no. 3, pp. 1301–1314, 2015.
- [32] <https://stemcellthailand.org/neurons-definition-function-neurotransmitters/>.
- [33] M. Defoort, T. Floquet, A. Kökösy, and W. Perruquetti, "Sliding-mode formation control for cooperative autonomous mobile robots," *IEEE Transactions on Industrial Electronics*, vol. 55, no. 11, pp. 3944–3953, 2008.
- [34] X. Liu, C. Yang, Z. Chen, M. Wang, and C. Su, "Neuro-adaptive observer based control of flexible joint robot," *Neurocomputing*, 2017.
- [35] F. Hamerlain, T. Floquet, and W. Perruquetti, "Experimental tests of a sliding mode controller for trajectory tracking of a car-like mobile robot," *Robotica*, vol. 32, no. 1, pp. 63–76, 2014.
- [36] R. J. de Jesús, "Discrete time control based in neural networks for pendulums," *Applied Soft Computing*, 2017.
- [37] Y. Pan, M. J. Er, T. Sun, B. Xu, and H. Yu, "Adaptive fuzzy PD control with stable H_{∞} tracking guarantee," *Neurocomputing*, vol. 237, pp. 71–78, 2017.
- [38] R. J. de Jesús, "Adaptive least square control in discrete time of robotic arms," *Soft Computing*, vol. 19, no. 12, pp. 3665–3676, 2015.
- [39] S. Commuri, S. Jagannathan, and F. L. Lewis, "CMAC neural network control of robot manipulators," *Journal of Robotic Systems*, vol. 14, no. 6, pp. 465–482, 1997.
- [40] J. S. Albus, "Theoretical and experimental aspects of a cerebellar model," *Developmental Disabilities Research Reviews*, vol. 17, pp. 93–101, 1972.
- [41] B. Yang, R. Bao, and H. Han, "Robust hybrid control based on PD and novel CMAC with improved architecture and learning scheme for electric load simulator," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 10, pp. 5271–5279, 2014.
- [42] S. Jagannathan and F. L. Lewis, "Multilayer discrete-time neural-net controller with guaranteed performance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 7, no. 1, pp. 107–130, 1996.
- [43] S. S. Ge and J. Wang, "Robust adaptive neural control for a class of perturbed strict feedback nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 13, no. 6, pp. 1409–1419, 2002.
- [44] Y. H. Kim, F. L. Lewis, and C. T. Abdallah, "A dynamic recurrent neural-network-based adaptive observer for a class of nonlinear systems," *Automatica*, vol. 33, no. 8, pp. 1539–1543, 1997.
- [45] J.-Q. Huang and F. L. Lewis, "Neural-network predictive control for nonlinear dynamic systems with time-delay," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 14, no. 2, pp. 377–389, 2003.
- [46] S. S. Ge, F. Hong, and T. H. Lee, "Adaptive neural control of nonlinear time-delay systems with unknown virtual control coefficients," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 1, pp. 499–516, 2004.
- [47] J. Na, X. M. Ren, and H. Huang, "Time-delay positive feedback control for nonlinear time-delay systems with neural network compensation," *Zidonghua Xuebao/Acta Automatica Sinica*, vol. 34, no. 9, pp. 1196–1202, 2008.
- [48] J. Na, X. Ren, C. Shang, and Y. Guo, "Adaptive neural network predictive control for nonlinear pure feedback systems with input delay," *Journal of Process Control*, vol. 22, no. 1, pp. 194–206, 2012.
- [49] R. R. Selmic and F. L. Lewis, "Neural-network approximation of piecewise continuous functions: application to friction compensation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 13, no. 3, pp. 745–751, 2002.
- [50] H. Zhang and F. L. Lewis, "Adaptive cooperative tracking control of higher-order nonlinear systems with unknown dynamics," *Automatica*, vol. 48, no. 7, pp. 1432–1439, 2012.
- [51] J. Na, X. Ren, and D. Zheng, "Adaptive control for nonlinear pure-feedback systems with high-order sliding mode observer," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 3, pp. 370–382, 2013.
- [52] J. Na, Q. Chen, X. Ren, and Y. Guo, "Adaptive prescribed performance motion control of servo mechanisms with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 486–494, 2014.
- [53] J. Na, X. Ren, G. Herrmann, and Z. Qiao, "Adaptive neural dynamic surface control for servo systems with unknown dead-zone," *Control Engineering Practice*, vol. 19, no. 11, pp. 1328–1343, 2011.
- [54] G. Li, J. Na, D. P. Stoten, and X. Ren, "Adaptive neural network feedforward control for dynamically substructured systems," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 944–954, 2014.

- [55] B. Xu, Z. Shi, C. Yang, and F. Sun, "Composite neural dynamic surface control of a class of uncertain nonlinear systems in strict-feedback form," *IEEE Transactions on Cybernetics*, 2014.
- [56] M. Chen and S. Ge, "Adaptive neural output feedback control of uncertain nonlinear systems with unknown hysteresis using disturbance observer," *IEEE Transactions on Industrial Electronics*, 2015.
- [57] M. Chen and S. S. Ge, "Direct adaptive neural control for a class of uncertain nonaffine nonlinear systems based on disturbance observer," *IEEE Transactions on Cybernetics*, vol. 43, no. 4, pp. 1213–1225, 2013.
- [58] M. Chen, G. Tao, and B. Jiang, "Dynamic surface control using neural networks for a class of uncertain nonlinear systems with input saturation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 2086–2097, 2015.
- [59] P. J. Werbos, "Approximate dynamic programming for real-time control and neural modeling," in *Handbook of Intelligent Control*, 1992.
- [60] F.-Y. Wang, H. Zhang, and D. Liu, "Adaptive dynamic programming: an introduction," *IEEE Computational Intelligence Magazine*, vol. 4, no. 2, pp. 39–47, 2009.
- [61] F. L. Lewis, D. Vrabie, and K. . Vamvoudakis, "Reinforcement learning and feedback control: using natural decision methods to design optimal adaptive controllers," *IEEE Control Systems Magazine*, vol. 32, no. 6, pp. 76–105, 2012.
- [62] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits and Systems Magazine*, vol. 9, no. 3, pp. 32–50, 2009.
- [63] A. Al-Tamimi, F. L. Lewis, and M. Abu-Khalaf, "Discrete-time nonlinear HJB solution using approximate dynamic programming: convergence proof," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 4, pp. 943–949, 2008.
- [64] H. Zhang, Y. Luo, and D. Liu, "Neural-network-based near-optimal control for a class of discrete-time affine nonlinear systems with control constraints," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 20, no. 9, pp. 1490–1503, 2009.
- [65] D. Wang, D. Liu, Q. Wei, D. Zhao, and N. Jin, "Optimal control of unknown nonaffine nonlinear discrete-time systems based on adaptive dynamic programming," *Automatica*, vol. 48, no. 8, pp. 1825–1832, 2012.
- [66] H. He, Z. Ni, and J. Fu, "A three-network architecture for online learning and optimization based on adaptive dynamic programming," *Neurocomputing*, vol. 78, no. 1, pp. 3–13, 2012.
- [67] D. Liu and Q. Wei, "Policy iteration adaptive dynamic programming algorithm for discrete-time nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 621–634, 2014.
- [68] D. Liu, X. Yang, D. Wang, and Q. Wei, "Reinforcement-learning-based robust controller design for continuous-time uncertain nonlinear systems subject to input constraints," *IEEE Transactions on Cybernetics*, vol. 45, no. 7, pp. 1372–1385, 2015.
- [69] J. Fu, H. He, and X. Zhou, "Adaptive learning and control for MIMO system based on adaptive dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 22, no. 7, pp. 1133–1148, 2011.
- [70] Y. Lv, J. Na, Q. Yang, X. Wu, and Y. Guo, "Online adaptive optimal control for continuous-time nonlinear systems with completely unknown dynamics," *International Journal of Control*, vol. 89, no. 1, pp. 99–112, 2016.
- [71] J. Na and G. Herrmann, "Online adaptive approximate optimal tracking control with simplified dual approximation structure for continuous-time unknown nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 1, no. 4, pp. 412–422, 2014.
- [72] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.
- [73] S. F. Ding, H. Li, C. Y. Su, J. Z. Yu, and F. X. Jin, "Evolutionary artificial neural networks: a review," *Artificial Intelligence Review*, vol. 39, no. 3, pp. 251–260, 2013.
- [74] X. Yao and Y. Liu, "A new evolutionary system for evolving artificial neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 8, no. 3, pp. 694–713, 1997.
- [75] Y. Li and A. Häußler, "Artificial evolution of neural networks and its application to feedback control," *Artificial Intelligence in Engineering*, vol. 10, no. 2, pp. 143–152, 1996.
- [76] C.-K. Goh, E.-J. Teoh, and K. C. Tan, "Hybrid multiobjective evolutionary design for artificial neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 19, no. 9, pp. 1531–1548, 2008.
- [77] K. C. Tan and Y. Li, "Grey-box model identification via evolutionary computing," *Control Engineering Practice*, vol. 10, no. 7, pp. 673–684, 2002.
- [78] L. Wang, C. K. Tan, and C. M. Chew, *Evolutionary robotics: from algorithms to implementations*, Chew C M. Evolutionary robotics, from algorithms to implementations[M]. NJ, 2006.
- [79] J. Zhang, Z.-H. Zhang, Y. Lin et al., "Evolutionary computation meets machine learning: a survey," *IEEE Computational Intelligence Magazine*, vol. 6, no. 4, pp. 68–75, 2011.
- [80] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, 2017.
- [81] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, 2017.
- [82] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, 2014.
- [83] C. Yang, T. Teng, B. Xu, Z. Li, J. Na, and C. Su, "Global adaptive tracking control of robot manipulators using neural networks with finite-time learning convergence," *International Journal of Control, Automation, and Systems*, vol. 15, no. 4, pp. 1916–1924, 2017.
- [84] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, 2017.
- [85] R. Cui and W. Yan, "Mutual synchronization of multiple robot manipulators with unknown dynamics," *Journal of Intelligent & Robotic Systems*, vol. 68, no. 2, pp. 105–119, 2012.
- [86] L. Cheng, Z.-G. Hou, M. Tan, and W. J. Zhang, "Tracking control of a closed-chain five-bar robot with two degrees of freedom by integration of an approximation-based approach and mechanical design," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 5, pp. 1470–1479, 2012.
- [87] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.

- [88] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.
- [89] L. Cheng, Z.-G. Hou, and M. Tan, "Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model," *Automatica*, vol. 45, no. 10, pp. 2312–2318, 2009.
- [90] W. He, Y. Dong, and C. Sun, "Adaptive Neural Impedance Control of a Robotic Manipulator with Input Saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334–344, 2016.
- [91] W. He, A. O. David, Z. Yin, and C. Sun, "Neural network control of a robotic manipulator with input deadzone and output constraint," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2015.
- [92] W. He, Z. Yin, and C. Sun, "Adaptive Neural Network Control of a Marine Vessel With Constraints Using the Asymmetric Barrier Lyapunov Function," *IEEE Transactions on Cybernetics*, 2016.
- [93] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.
- [94] C. Sun, W. He, and J. Hong, "Neural Network Control of a Flexible Robotic Manipulator Using the Lumped Spring-Mass Model," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 1863–1874, 2017.
- [95] W. He, Y. Ouyang, and J. Hong, "Vibration Control of a Flexible Robotic Manipulator in the Presence of Input Deadzone," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 48–59, 2017.
- [96] R. Cui, X. Zhang, and D. Cui, "Adaptive sliding-mode attitude control for autonomous underwater vehicles with input nonlinearities," *Ocean Engineering*, vol. 123, pp. 45–54, 2016.
- [97] R. Cui, C. Yang, Y. Li, and S. Sharma, "Adaptive Neural Network Control of AUVs With Control Input Nonlinearities Using Reinforcement Learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 6, pp. 1019–1029, 2017.
- [98] B. Xu, D. Wang, Y. Zhang, and Z. Shi, "DOB based neural control of flexible hypersonic flight vehicle considering wind effects," *IEEE Transactions on Industrial Electronics*, vol. PP, no. 99, p. 1, 2017.
- [99] B. Xu, C. Yang, and Y. Pan, "Global neural dynamic surface tracking control of strict-feedback systems with application to hypersonic flight vehicle," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2563–2575, 2015.
- [100] Y. Li, S. S. Ge, and C. Yang, "Learning impedance control for physical robot-environment interaction," *International Journal of Control*, vol. 85, no. 2, pp. 182–193, 2012.
- [101] Y. Li, S. S. Ge, Q. Zhang, and T. . Lee, "Neural networks impedance control of robots interacting with environments," *IET Control Theory & Applications*, vol. 7, no. 11, pp. 1509–1519, 2013.
- [102] Y. Li and S. S. Ge, "Impedance learning for robots interacting with unknown environments," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 4, pp. 1422–1432, 2014.
- [103] C. Wang, Y. Li, S. S. Ge, and T. . Lee, "Optimal critic learning for robot control in time-varying environments," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 10, pp. 2301–2310, 2015.
- [104] Y. Li, L. Chen, K. P. Tee, and Q. Li, "Reinforcement learning control for coordinated manipulation of multi-robots," *Neurocomputing*, vol. 170, pp. 168–175, 2015.
- [105] Y. Li and S. S. Ge, "Human—robot collaboration based on motion intention estimation," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 3, pp. 1007–1014, 2013.
- [106] Y. Li, K. P. Tee, W. L. Chan, R. Yan, Y. Chua, and D. K. Limbu, "Continuous Role Adaptation for Human-Robot Shared Control," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 672–681, 2015.
- [107] Y. Li, K. P. Tee, R. Yan, W. L. Chan, and Y. Wu, "A Framework of Human-Robot Coordination Based on Game Theory and Policy Iteration," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1408–1418, 2016.
- [108] A. Clark, *Surfing uncertainty: Prediction, action, and the embodied mind*, Oxford University Press, 2015.
- [109] J. Zhong, C. Weber, and S. Wermter, "Learning features and predictive transformation encoding based on a horizontal product model," *Neural Networks and Machine Learning ICANN*, pp. 539–546, 2012.
- [110] J. Zhong, C. Weber, and S. Wermter, "A predictive network architecture for a robust and smooth robot docking behavior," *Journal of Behavioral Robotics*, vol. 3, no. 4, pp. 172–180, 2012.
- [111] W. Prinz, "Perception and Action Planning," *European Journal of Cognitive Psychology*, vol. 9, no. 2, pp. 129–154, 1997.
- [112] J. Zhong, M. Peniak, J. Tani, T. Ogata, and A. Cangelosi, "Sensorimotor Input as a Language Generalisation Tool: A Neurorobotics Model for Generation and Generalisation of Noun-Verb Combinations with Sensorimotor Inputs," Tech. Rep., arXiv, 1605.03261, 2016, arXiv:1605.03261.
- [113] H. T. Siegelmann and E. D. Sontag, "On the computational power of neural nets," *Journal of Computer and System Sciences*, vol. 50, no. 1, pp. 132–150, 1995.
- [114] J. Zhong, Artificial Neural Models for Feedback Pathways for Sensorimotor Integration.
- [115] J. Tani, M. Ito, and Y. Sugita, "Self-organization of distributedly represented multiple behavior schemata in a mirror system: Reviews of robot experiments using RNNPB," *Neural Networks*, vol. 17, no. 8–9, pp. 1273–1289, 2004.
- [116] W. Hinoshita, H. Arie, J. Tani, H. G. Okuno, and T. Ogata, "Emergence of hierarchical structure mirroring linguistic composition in a recurrent neural network," *Neural Networks*, vol. 24, no. 4, pp. 311–320, 2011.
- [117] J. Tani, *Exploring robotic minds: actions, symbols, and consciousness as self-organizing dynamic phenomena*, Oxford University Press, 2016.
- [118] A. Ahmadi and J. Tani, "How can a recurrent neurodynamic predictive coding model cope with fluctuation in temporal patterns? Robotic experiments on imitative interaction," *Neural Networks*, vol. 92, pp. 3–16, 2017.
- [119] J. Zhong, A. Cangelosi, and S. Wermter, "Toward a self-organizing pre-symbolic neural model representing sensorimotor primitives," *Frontiers in Behavioral Neuroscience*, vol. 8, article no. 22, 2014.
- [120] H. K. Abbas, R. M. Zablotowicz, and H. A. Bruns, "Modeling the colonization of maize by toxigenic and non-toxicogenic *Aspergillus flavus* strains: implications for biological control," *World Mycotoxin Journal*, vol. 1, no. 3, pp. 333–340, 2008.
- [121] J. Zhong and L. Canamero, "From continuous affective space to continuous expression space: Non-verbal behaviour recognition and generation," in *Proceedings of the 4th Joint IEEE*

- International Conference on Development and Learning and on Epigenetic Robotics, IEEE ICDL-EPIROB 2014*, pp. 75–80, ita, October 2014.
- [122] J. Zhong, A. Cangelosi, and T. Ogata, “Toward Abstraction from Multi-modal Data: Empirical Studies on Multiple Time-scale Recurrent Models,” in *Proceedings of the International Joint Conference on Artificial Neural Networks (IJCNN)*, 2017.
- [123] G. Park and J. Tani, “Development of compositional and contextual communicable congruence in robots by using dynamic neural network models,” *Neural Networks*, vol. 72, pp. 109–122, 2015.
- [124] A. Ahmadi and J. Tani, “Bridging the gap between probabilistic and deterministic models: a simulation study on a variational Bayes predictive coding recurrent neural network model,” 2017, <https://arxiv.org/abs/1706.10240>.
- [125] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
- [126] J. Schmidhuber, “Deep learning in neural networks: an overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015.
- [127] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.
- [128] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, “A survey of deep neural network architectures and their applications,” *Neurocomputing*, vol. 234, pp. 11–26, 2017.
- [129] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,” *American Association for the Advancement of Science: Science*, vol. 313, no. 5786, pp. 504–507, 2006.
- [130] B. C. Pijanowski, A. Tayyebi, J. Doucette, B. K. Pekin, D. Braun, and J. Plourde, “A big data urban growth simulation at a national scale: configuring the GIS and neural network based Land Transformation Model to run in a High Performance Computing (HPC) environment,” *Environmental Modelling & Software*, vol. 51, pp. 250–268, 2014.
- [131] D. C. Cireřan, U. Meier, L. M. Gambardella, and J. Schmidhuber, “Deep, big, simple neural nets for handwritten digit recognition,” *Neural Computation*, vol. 22, no. 12, pp. 3207–3220, 2010.

Research Article

The Hierarchical Iterative Identification Algorithm for Multi-Input-Output-Error Systems with Autoregressive Noise

Jiling Ding

Department of Mathematics, Jining University, Qufu 273155, China

Correspondence should be addressed to Jiling Ding; jlding@jnxu.edu.cn

Received 15 July 2017; Accepted 18 September 2017; Published 29 October 2017

Academic Editor: Guang Li

Copyright © 2017 Jiling Ding. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper considers the identification problem of multi-input-output-error autoregressive systems. A hierarchical gradient based iterative (H-GI) algorithm and a hierarchical least squares based iterative (H-LSI) algorithm are presented by using the hierarchical identification principle. A gradient based iterative (GI) algorithm and a least squares based iterative (LSI) algorithm are presented for comparison. The simulation results indicate that the H-LSI algorithm can obtain more accurate parameter estimates than the LSI algorithm, and the H-GI algorithm converges faster than the GI algorithm.

1. Introduction

System identification studies mathematical models of dynamic systems by fitting experimental data to a suitable model structure [1, 2]. Many practical systems have multiple inputs and multiple outputs such as chemical processes [3, 4], automation devices [5–7], and network communication engineering [8–10]. For decades, much research has been performed on the multivariable systems [11, 12], and some typical approaches for the parameter estimation of the multivariable systems have been reported [13], such as the canonical approach [14], the iterative methods [15, 16], and the least squares methods [17]. Recently, Panda and Vijayaraghavan adopted the sequential relay feedback test to estimate the parameter of the linear multivariable systems [18]. Jafari et al. presented an iterative least squares algorithm to identify the multivariable nonlinear systems with colored noises [19].

The multivariable systems contain both parameter vectors and parameter matrices, and the systems inputs and system outputs are relevant and coupled [20–22]. For the sake of reducing the computational complexity, the hierarchical identification principle is utilized to transform a complex system into several subsystems and then to estimate the parameter vector of each subsystem [23, 24], respectively. In this literature, Schranz et al. proposed a feasible hierarchical

identification process for identifying the viscoelastic model of respiratory mechanics [25]. Xu et al. developed the parameter estimation for dynamical response signals [26, 27].

The iterative methods have been widely applied in identifying the parameters of linear or nonlinear systems [28–30]. Many iterative algorithms for system identification are based on the gradient method [31] and the least squares method [32–35]. The basic idea of iterative methods is to update the parameter estimates using batch data.

This paper focuses on the parameter estimation for output-error autoregressive (OEAR) systems using the hierarchical identification principle and the iterative identification principle and presents a hierarchical gradient based iterative (H-GI) algorithm and a hierarchical least squares based iterative (H-LSI) algorithm. The key is to decompose a multi-input OEAR system into two subsystems and then to identify each subsystem. The work in [36, 37] discussed the single-input single-output systems, but many practical systems have multiple inputs and multiple outputs with the development of industrial technology. Compared with the work in [36, 37], this paper discusses the parameter estimation for multi-input OEAR systems and the presented H-LSI algorithm can achieve higher estimation accuracy than the LSI algorithm, and the H-GI algorithm also can achieve higher estimation accuracy than the GI algorithm.

The rest of this paper is organized as follows. Section 2 gives some definitions and the identification model of multi-input OEAR systems. Section 3 presents a gradient based iterative algorithm and a least squares based iterative algorithm for multi-input OEAR systems. Section 4 derives a hierarchical gradient based iterative algorithm. Section 5 derives a hierarchical least squares based iterative algorithm. Section 6 provides two illustrative examples to demonstrate the effectiveness of the proposed algorithms. Finally, concluding remarks are given in Section 7.

2. The Problem Formulation

Let us define some notation.

Symbols: meaning

$\mathbf{1}_n$: an n dimensional column vector whose entries are all 1

p_0 : a large positive constant, for example, $p_0 = 10^6$

\mathbf{X}^T : the transpose of the vector or matrix \mathbf{X}

$\|\mathbf{X}\|^2$: $\|\mathbf{X}\|^2 = \text{tr}[\mathbf{X}\mathbf{X}^T]$

$X := A$: A defined as X

$\lambda_{\max}[\mathbf{X}]$: the maximum eigenvalue of the symmetric real matrix \mathbf{X} .

Consider the following multi-input-output-error type models:

$$y(\tau) = \sum_{j=1}^r \frac{B_j(z)}{A_j(z)} u_j(\tau) + w(\tau), \quad (1)$$

where $y(\tau) \in \mathbb{R}$ is the system output, $u_j(\tau) \in \mathbb{R}$, $j = 1, 2, \dots, r$, are the system inputs, and $w(\tau) \in \mathbb{R}$ is the colored noise with zero mean. $A_j(z)$ and $B_j(z)$ are polynomials in the unit backward shift operator z^{-1} , and

$$A_j(z) := 1 + a_{j1}z^{-1} + a_{j2}z^{-2} + \dots + a_{jn_j}z^{-n_j}, \quad a_{ji} \in \mathbb{R}, \quad (2)$$

$$B_j(z) := b_{j1}z^{-1} + b_{j2}z^{-2} + \dots + b_{jn_j}z^{-n_j}, \quad b_{ji} \in \mathbb{R}.$$

Assume that the orders n_j are known, $y(\tau) = 0$, $u_j(\tau) = 0$, and $w(\tau) = 0$ as $\tau \leq 0$. The colored noise $w(\tau)$ can be fitted by a moving average process

$$w(\tau) = D(z)v(\tau), \quad (3)$$

or an autoregressive process

$$w(\tau) = \frac{1}{C(z)}v(\tau), \quad (4)$$

or an autoregressive moving average process

$$w(\tau) = \frac{D(z)}{C(z)}v(\tau), \quad (5)$$

where $v(\tau)$ is the white noise with zero mean and $C(z)$, $D(z)$ are polynomials in the unit backward shift operator z^{-1} :

$$C(z) := 1 + c_1z^{-1} + c_2z^{-2} + \dots + c_{n_c}z^{-n_c}, \quad c_j \in \mathbb{R}, \quad (6)$$

$$D(z) := 1 + d_1z^{-1} + d_2z^{-2} + \dots + d_{n_d}z^{-n_d}, \quad d_j \in \mathbb{R}.$$

This paper considers the colored noise to be an autoregressive process, so the models in (1) can be taken as the multi-input OEAR systems.

Define the intermediate variables:

$$x_j(\tau) := \frac{B_j(z)}{A_j(z)} u_j(\tau), \quad j = 1, 2, \dots, r. \quad (7)$$

From (4) and (7), we have

$$\begin{aligned} w(\tau) &= [1 - C(z)]w(\tau) + v(\tau) \\ &= -\sum_{i=1}^{n_c} c_i w(\tau - i) + v(\tau), \end{aligned} \quad (8)$$

$$\begin{aligned} x_j(\tau) &= [1 - A_j(z)]x_j(\tau) + B_j(z)u_j(\tau) \\ &= \sum_{i=1}^{n_j} [-a_{ji}x_j(\tau - i) + b_{ji}u_j(\tau - i)]. \end{aligned}$$

The output $y(\tau)$ in (1) can be written as

$$\begin{aligned} y(\tau) &= \sum_{j=1}^r \sum_{i=1}^{n_j} [-a_{ji}x_j(\tau - i) + b_{ji}u_j(\tau - i)] \\ &\quad - \sum_{i=1}^{n_c} c_i w(\tau - i) + v(\tau). \end{aligned} \quad (9)$$

Define the parameter vectors as

$$\boldsymbol{\theta} := [\boldsymbol{\vartheta}^T, \mathbf{c}^T]^T \in \mathbb{R}^n, \quad n := n_0 + n_c, \quad n_0 := \sum_{j=1}^r 2n_j,$$

$$\boldsymbol{\vartheta} := [\boldsymbol{\vartheta}_1^T, \boldsymbol{\vartheta}_2^T, \dots, \boldsymbol{\vartheta}_r^T]^T \in \mathbb{R}^{n_0},$$

$$\boldsymbol{\vartheta}_j := [a_{j1}, a_{j2}, \dots, a_{jn_j}, b_{j1}, b_{j2}, \dots, b_{jn_j}]^T \in \mathbb{R}^{2n_j}, \quad (10)$$

$$j = 1, 2, \dots, r,$$

$$\mathbf{c} := [c_1, c_2, \dots, c_{n_c}]^T \in \mathbb{R}^{n_c}$$

and the information vectors as

$$\begin{aligned}\boldsymbol{\varphi}(\tau) &:= [\boldsymbol{\phi}^T(\tau), \boldsymbol{\psi}^T(\tau)]^T \in \mathbb{R}^n, \\ \boldsymbol{\phi}(\tau) &:= [\boldsymbol{\phi}_1^T(\tau), \boldsymbol{\phi}_2^T(\tau), \dots, \boldsymbol{\phi}_r^T(\tau)]^T \in \mathbb{R}^{n_0}, \\ \boldsymbol{\phi}_j(\tau) &:= [-x_j(\tau-1), -x_j(\tau-2), \dots, \\ &\quad -x_j(\tau-n_j), u_j(\tau-1), u_j(\tau-2), \dots, u_j(\tau-n_j)]^T \quad (11) \\ &\in \mathbb{R}^{2n_j}, \\ \boldsymbol{\psi}(\tau) &:= [-w(\tau-1), -w(\tau-2), \dots, -w(\tau-n_c)]^T \\ &\in \mathbb{R}^{n_c}.\end{aligned}$$

According to the above definitions, (8) and (9) can be written as

$$w(\tau) = \boldsymbol{\psi}^T(\tau) \mathbf{c} + v(\tau), \quad (12)$$

$$x_j(\tau) = \boldsymbol{\phi}_j^T(\tau) \boldsymbol{\vartheta}_j, \quad (13)$$

$$\begin{aligned}y(\tau) &= \sum_{j=1}^r \boldsymbol{\phi}_j^T(\tau) \boldsymbol{\vartheta}_j + \boldsymbol{\psi}^T(\tau) \mathbf{c} + v(\tau) \\ &= \boldsymbol{\varphi}^T(\tau) \boldsymbol{\theta} + v(\tau).\end{aligned} \quad (14)$$

Equation (14) is the identification model of the multi-input OEAR system.

3. The Gradient Based and Least Squares Based Iterative Algorithm

Consider the data $\{u_1(i), u_2(i), \dots, u_r(i), y(i)\}$ from $i = \tau - p + 1$ to $i = \tau$ and define quadratic criterion function as

$$J(\boldsymbol{\theta}) := \sum_{i=\tau-p+1}^{\tau} [y(i) - \boldsymbol{\varphi}^T(i) \boldsymbol{\theta}]^2. \quad (15)$$

Let $k = 1, 2, 3, \dots$ be an iteration variable and $\hat{\boldsymbol{\theta}}_k(\tau)$ be the estimate of $\boldsymbol{\theta}$ at iteration k . Minimizing $J(\boldsymbol{\theta})$ by using the negative gradient search, we can obtain

$$\begin{aligned}\hat{\boldsymbol{\theta}}_k(\tau) &= \hat{\boldsymbol{\theta}}_{k-1}(\tau) - \frac{\mu_k(\tau)}{2} \text{grad} [J(\hat{\boldsymbol{\theta}}_{k-1}(\tau))] \\ &= \hat{\boldsymbol{\theta}}_{k-1}(\tau) \\ &\quad + \mu_k(\tau) \sum_{i=\tau-p+1}^{\tau} \boldsymbol{\varphi}(i) [y(i) - \boldsymbol{\varphi}^T(i) \hat{\boldsymbol{\theta}}_{k-1}(\tau)],\end{aligned} \quad (16)$$

where $\mu_k(\tau) > 0$ is an iterative step-size. Because the information vector $\boldsymbol{\varphi}(\tau)$ contains the unknown variables $x_j(\tau-i)$ and $w(\tau-i)$, we use the estimates $\hat{x}_{j,k-1}(\tau-i)$ and $\hat{w}_{k-1}(\tau-i)$ at iteration $k-1$ to replace the unknown variables $x_j(\tau-i)$ and $w(\tau-i)$; we can obtain the gradient based iterative

(GI) algorithm for estimating the parameter vector $\boldsymbol{\theta}$ of the multi-input OEAR systems:

$$\begin{aligned}\hat{\boldsymbol{\theta}}_k(\tau) &= \hat{\boldsymbol{\theta}}_{k-1}(\tau-1) + \mu_k \sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\varphi}}_k(i) [y(i) - \hat{\boldsymbol{\varphi}}_k^T(i) \\ &\quad \cdot \hat{\boldsymbol{\theta}}_{k-1}(\tau)], \quad k = 1, 2, \dots, \\ \hat{\boldsymbol{\varphi}}_k(\tau) &= [\hat{\boldsymbol{\varphi}}_k^T(\tau), \hat{\boldsymbol{\psi}}_k^T(\tau)]^T, \\ \hat{\boldsymbol{\varphi}}_k(\tau) &= [\hat{\boldsymbol{\phi}}_{1,k}^T(\tau), \hat{\boldsymbol{\phi}}_{2,k}^T(\tau), \dots, \hat{\boldsymbol{\phi}}_{r,k}^T(\tau)]^T, \\ \hat{\boldsymbol{\varphi}}_{j,k}(\tau) &= [-\hat{x}_{j,k-1}(\tau-1), \dots, -\hat{x}_{j,k-1}(\tau-n_j), \\ &\quad u_j(\tau-1), \dots, u_j(\tau-n_j)]^T, \\ \hat{\boldsymbol{\psi}}_k(\tau) &= [-\hat{w}_{k-1}(\tau-1), -\hat{w}_{k-1}(\tau-2), \dots, \\ &\quad -\hat{w}_{k-1}(\tau-n_c)]^T, \\ \hat{x}_{j,k}(\tau-i) &= \hat{\boldsymbol{\phi}}_{j,k}^T(\tau-i) \hat{\boldsymbol{\vartheta}}_{j,k}(\tau), \\ \hat{w}_k(\tau-i) &= y(\tau-i) - \hat{\boldsymbol{\varphi}}_k^T(\tau-i) \hat{\boldsymbol{\theta}}_k(\tau), \\ \mu_k &\leq 2\lambda_{\max}^{-1} \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\varphi}}_k(i) \hat{\boldsymbol{\varphi}}_k^T(i) \right], \\ \hat{\boldsymbol{\theta}}_k(\tau) &= [\hat{\boldsymbol{\vartheta}}_k^T(\tau), \hat{\mathbf{c}}_k^T(\tau)]^T, \\ \hat{\boldsymbol{\vartheta}}_k(\tau) &= [\hat{\boldsymbol{\vartheta}}_{1,k}^T(\tau), \hat{\boldsymbol{\vartheta}}_{2,k}^T(\tau), \dots, \hat{\boldsymbol{\vartheta}}_{r,k}^T(\tau)]^T, \\ \hat{\boldsymbol{\vartheta}}_{j,k}(\tau) &= [\hat{a}_{j1,k}(\tau), \hat{a}_{j2,k}(\tau), \dots, \hat{a}_{jn_j,k}(\tau), \hat{b}_{j1,k}(\tau), \\ &\quad \hat{b}_{j2,k}(\tau), \dots, \hat{b}_{jn_j,k}(\tau)]^T, \\ \hat{\mathbf{c}}_k(\tau) &= [\hat{c}_{1,k}(\tau), \hat{c}_{2,k}(\tau), \dots, \hat{c}_{n_c,k}(\tau)]^T.\end{aligned} \quad (17)$$

The convergence rate of the GI algorithm is slow. To improve the convergence speed, we derive a least squares based iterative (LSI) identification algorithm. Minimizing $J(\boldsymbol{\theta})$ and letting the derivative of $J(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ be zero give the LSI identification algorithm for the multi-input OEAR systems:

$$\begin{aligned}\hat{\boldsymbol{\theta}}_k(\tau) &= \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\varphi}}_k(i) \hat{\boldsymbol{\varphi}}_k^T(i) \right]^{-1} \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\varphi}}_k(i) y(i) \right], \\ &\quad k = 1, 2, 3, \dots,\end{aligned}$$

$$\hat{\boldsymbol{\varphi}}_k(\tau) = [\hat{\boldsymbol{\phi}}_k^T(\tau), \hat{\boldsymbol{\psi}}_k^T(\tau)]^T,$$

$$\hat{\boldsymbol{\phi}}_k(\tau) = [\hat{\boldsymbol{\phi}}_{1,k}^T(\tau), \hat{\boldsymbol{\phi}}_{2,k}^T(\tau), \dots, \hat{\boldsymbol{\phi}}_{r,k}^T(\tau)]^T,$$

$$\begin{aligned}
\widehat{\boldsymbol{\phi}}_{j,k}(\tau) &= \left[-\widehat{x}_{j,k-1}(\tau-1), \dots, -\widehat{x}_{j,k-1}(\tau-n_j), \right. \\
&\quad \left. u_j(\tau-1), \dots, u_j(\tau-n_j) \right]^T, \\
\widehat{\boldsymbol{\psi}}_k(\tau) &= \left[-\widehat{w}_{k-1}(\tau-1), -\widehat{w}_{k-1}(\tau-2), \dots, \right. \\
&\quad \left. -\widehat{w}_{k-1}(\tau-n_c) \right]^T, \\
\widehat{x}_{j,k}(\tau-i) &= \widehat{\boldsymbol{\phi}}_{j,k}^T(\tau-i) \widehat{\boldsymbol{\theta}}_{j,k}(\tau), \\
\widehat{w}_k(\tau-i) &= y(\tau-i) - \widehat{\boldsymbol{\phi}}_k^T(\tau-i) \widehat{\boldsymbol{\theta}}_k(\tau), \\
\widehat{\boldsymbol{\theta}}_k(\tau) &= \left[\widehat{\boldsymbol{\theta}}_k^T(\tau), \widehat{\mathbf{c}}_k^T(\tau) \right]^T, \\
\widehat{\boldsymbol{\theta}}_k(\tau) &= \left[\widehat{\boldsymbol{\theta}}_{1,k}^T(\tau), \widehat{\boldsymbol{\theta}}_{2,k}^T(\tau), \dots, \widehat{\boldsymbol{\theta}}_{r,k}^T(\tau) \right]^T, \\
\widehat{\boldsymbol{\theta}}_{j,k}(\tau) &= \left[\widehat{a}_{j1,k}(\tau), \widehat{a}_{j2,k}(\tau), \dots, \widehat{a}_{jn_j,k}(\tau), \widehat{b}_{j1,k}(\tau), \right. \\
&\quad \left. \widehat{b}_{j2,k}(\tau), \dots, \widehat{b}_{jm_j,k}(\tau) \right]^T, \\
\widehat{\mathbf{c}}_k(\tau) &= \left[\widehat{c}_{1,k}(\tau), \widehat{c}_{2,k}(\tau), \dots, \widehat{c}_{n_c,k}(\tau) \right]^T.
\end{aligned} \tag{18}$$

4. The Hierarchical Gradient Based Iterative Algorithm

Define intermediate variables:

$$y_1(\tau) := y(\tau) - \boldsymbol{\psi}^T(\tau) \mathbf{c}, \tag{19}$$

$$y_2(\tau) := y(\tau) - \boldsymbol{\phi}^T(\tau) \boldsymbol{\theta}. \tag{20}$$

Using the hierarchical identification principle, the multi-input OEAR system in (14) can be decomposed into two fictitious subsystems:

$$y_1(\tau) = \boldsymbol{\phi}^T(\tau) \boldsymbol{\theta} + v(\tau), \tag{21}$$

$$y_2(\tau) = \boldsymbol{\psi}^T(\tau) \mathbf{c} + v(\tau). \tag{22}$$

Next, we identify the parameters $\boldsymbol{\theta}$ and \mathbf{c} of each subsystem in (21) and (22), respectively. Define quadratic criterion functions as

$$J_1(\boldsymbol{\theta}, \mathbf{c}) := \sum_{i=\tau-p+1}^{\tau} \left[y_1(i) - \boldsymbol{\phi}^T(i) \boldsymbol{\theta} \right]^2, \tag{23}$$

$$J_2(\boldsymbol{\theta}, \mathbf{c}) := \sum_{i=\tau-p+1}^{\tau} \left[y_2(i) - \boldsymbol{\psi}^T(i) \mathbf{c} \right]^2.$$

Let $\widehat{\boldsymbol{\theta}}_k(\tau)$ and $\widehat{\mathbf{c}}_k(\tau)$ be the estimates of $\boldsymbol{\theta}$ and \mathbf{c} at iteration k . Using the negative gradient search and minimizing $J_1(\boldsymbol{\theta}, \mathbf{c})$ and $J_2(\boldsymbol{\theta}, \mathbf{c})$, we can obtain

$$\begin{aligned}
\widehat{\boldsymbol{\theta}}_k(\tau) &= \widehat{\boldsymbol{\theta}}_{k-1}(\tau) \\
&\quad + \mu_{1k}(\tau) \sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) \left[y_1(i) - \boldsymbol{\phi}^T(i) \widehat{\boldsymbol{\theta}}_{k-1}(\tau) \right], \\
\widehat{\mathbf{c}}_k(\tau) &= \widehat{\mathbf{c}}_{k-1}(\tau) \\
&\quad + \mu_{2k}(\tau) \sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) \left[y_2(i) - \boldsymbol{\psi}^T(i) \widehat{\mathbf{c}}_{k-1}(\tau) \right].
\end{aligned} \tag{24}$$

Here, $\mu_{1k}(\tau)$ and $\mu_{2k}(\tau)$ are the iterative step-sizes or convergence factors. Substituting (19) into (24) and (20) into (25), we can obtain

$$\begin{aligned}
\widehat{\boldsymbol{\theta}}_k(\tau) &= \widehat{\boldsymbol{\theta}}_{k-1}(\tau) + \mu_{1k}(\tau) \\
&\quad \cdot \sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) \left[y(i) - \boldsymbol{\psi}^T(i) \mathbf{c} - \boldsymbol{\phi}^T(i) \widehat{\boldsymbol{\theta}}_{k-1}(\tau) \right],
\end{aligned} \tag{26}$$

$$\begin{aligned}
\widehat{\mathbf{c}}_k(\tau) &= \widehat{\mathbf{c}}_{k-1}(\tau) + \mu_{2k}(\tau) \\
&\quad \cdot \sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) \left[y(i) - \boldsymbol{\phi}^T(i) \boldsymbol{\theta} - \boldsymbol{\psi}^T(i) \widehat{\mathbf{c}}_{k-1}(\tau) \right].
\end{aligned} \tag{27}$$

The parameter estimates $\widehat{\boldsymbol{\theta}}_k(\tau)$ and $\widehat{\mathbf{c}}_k(\tau)$ cannot be computed by (26) and (27), because the information vectors $\boldsymbol{\phi}(\tau)$ and $\boldsymbol{\psi}(\tau)$ contain unknown variables $x_j(\tau-i)$ and $w(\tau-i)$, and the parameter vectors $\boldsymbol{\theta}$ and \mathbf{c} in (26) and (27) are unknown. We solve this problem by replacing the unknown variables $x_j(\tau-i)$ and $w(\tau-i)$ with their corresponding estimates $\widehat{x}_{j,k-1}(\tau-i)$ and $\widehat{w}_{k-1}(\tau-i)$ at iteration $k-1$ and define the estimates $\widehat{\boldsymbol{\phi}}_k(\tau)$ and $\widehat{\boldsymbol{\psi}}_k(\tau)$ at iteration k as

$$\begin{aligned}
\widehat{\boldsymbol{\phi}}_k(\tau) &:= \left[\widehat{\boldsymbol{\phi}}_{1,k}^T(\tau), \widehat{\boldsymbol{\phi}}_{2,k}^T(\tau), \dots, \widehat{\boldsymbol{\phi}}_{r,k}^T(\tau) \right]^T \in \mathbb{R}^{n_0}, \\
\widehat{\boldsymbol{\phi}}_{j,k}(\tau) &:= \left[-\widehat{x}_{j,k-1}(\tau-1), \dots, \right. \\
&\quad \left. -\widehat{x}_{j,k-1}(\tau-n_j), u_j(\tau-1), \dots, u_j(\tau-n_j) \right]^T \\
&\quad \in \mathbb{R}^{2n_j}, \\
\widehat{\boldsymbol{\psi}}_k(\tau) &:= \left[-\widehat{w}_{k-1}(\tau-1), -\widehat{w}_{k-1}(\tau-2), \dots, \right. \\
&\quad \left. -\widehat{w}_{k-1}(\tau-n_c) \right]^T \in \mathbb{R}^{n_c}.
\end{aligned} \tag{28}$$

From (12) and (14), we have

$$\begin{aligned}
x_j(\tau-i) &= \boldsymbol{\phi}_j^T(\tau-i) \boldsymbol{\theta}_j, \\
w(\tau-i) &= y(\tau-i) - \boldsymbol{\phi}^T(\tau-i) \boldsymbol{\theta}.
\end{aligned} \tag{29}$$

Substituting $\phi_j(\tau-i)$ and ϑ_j with their estimates $\hat{\phi}_{j,k}(\tau-i)$ and $\hat{\vartheta}_{j,k}(\tau)$, we can get the estimates $\hat{x}_{j,k}(\tau-i)$ and $\hat{w}_k(\tau-i)$ at iteration k :

$$\begin{aligned}\hat{x}_{j,k}(\tau-i) &= \hat{\phi}_{j,k}^T(\tau-i) \hat{\vartheta}_{j,k}(\tau), \\ \hat{w}_k(\tau-i) &= y(\tau-i) - \hat{\phi}_k^T(\tau-i) \hat{\vartheta}_k(\tau).\end{aligned}\quad (30)$$

Replacing $\phi(\tau)$, $\psi(\tau)$ in (26) and (27) with their estimates $\hat{\phi}_k(\tau)$ and $\hat{\psi}_k(\tau)$, replacing \mathbf{c} in (26) with its estimate $\hat{\mathbf{c}}_{k-1}(\tau)$, and replacing ϑ in (27) with its estimate $\hat{\vartheta}_k(\tau)$, we have

$$\begin{aligned}\hat{\vartheta}_k(\tau) &= \hat{\vartheta}_{k-1}(\tau) + \mu_{1k}(\tau) \sum_{i=\tau-p+1}^{\tau} \hat{\phi}_k(i) \\ &\quad \cdot \left[y(i) - \hat{\phi}_k^T(i) \hat{\vartheta}_{k-1}(\tau) - \hat{\psi}_k^T(i) \hat{\mathbf{c}}_{k-1}(\tau) \right], \\ \hat{\mathbf{c}}_k(\tau) &= \hat{\mathbf{c}}_{k-1}(\tau) + \mu_{2k}(\tau) \sum_{i=\tau-p+1}^{\tau} \hat{\psi}_k(i) \\ &\quad \cdot \left[y(i) - \hat{\phi}_k^T(i) \hat{\vartheta}_k(\tau) - \hat{\psi}_k^T(i) \hat{\mathbf{c}}_{k-1}(\tau) \right].\end{aligned}\quad (31)$$

In order to guarantee the convergence of $\hat{\vartheta}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$, a conservative choice is

$$\begin{aligned}\mu_{1k}(\tau) &= \mu_{2k}(\tau) = \bar{\mu}_k(\tau) \\ &\leq \max \left[2\lambda_{\max}^{-1} \sum_{i=\tau-p+1}^{\tau} \hat{\phi}_k(i) \hat{\phi}_k^T(i), \right. \\ &\quad \left. 2\lambda_{\max}^{-1} \sum_{i=\tau-p+1}^{\tau} \hat{\psi}_k(i) \hat{\psi}_k^T(i) \right].\end{aligned}\quad (32)$$

At last, we can summarize the hierarchical gradient based iterative parameter estimation (H-GI) algorithm for estimating ϑ and \mathbf{c} of the multi-input OEAR systems:

$$\hat{\vartheta}_k(\tau) = \hat{\vartheta}_{k-1}(\tau) + \bar{\mu}_k(\tau) \sum_{i=\tau-p+1}^{\tau} \hat{\phi}_k(i) \left[y(i) - \hat{\phi}_k^T(i) \right. \quad (33)$$

$$\left. \cdot \hat{\vartheta}_{k-1}(\tau) - \hat{\psi}_k^T(i) \hat{\mathbf{c}}_{k-1}(\tau) \right],$$

$$\hat{\mathbf{c}}_k(\tau) = \hat{\mathbf{c}}_{k-1}(\tau) + \bar{\mu}_k(\tau) \sum_{i=\tau-p+1}^{\tau} \hat{\psi}_k(i) \left[y(i) - \hat{\phi}_k^T(i) \right. \quad (34)$$

$$\left. \cdot \hat{\vartheta}_k(\tau) - \hat{\psi}_k^T(i) \hat{\mathbf{c}}_{k-1}(\tau) \right],$$

$$\hat{\phi}_k(\tau) = \left[\hat{\phi}_{1,k}^T(\tau), \hat{\phi}_{2,k}^T(\tau), \dots, \hat{\phi}_{r,k}^T(\tau) \right]^T, \quad (35)$$

$$\hat{\phi}_{j,k}(\tau) = \left[-\hat{x}_{j,k-1}(\tau-1), \dots, -\hat{x}_{j,k-1}(\tau-n_j), u_j(\tau-1), \dots, u_j(\tau-n_j) \right]^T, \quad (36)$$

$$\hat{\psi}_k(\tau) = \left[-\hat{w}_{k-1}(\tau-1), -\hat{w}_{k-1}(\tau-2), \dots, -\hat{w}_{k-1}(\tau-n_c) \right]^T, \quad (37)$$

$$\hat{x}_{j,k}(\tau-i) = \hat{\phi}_{j,k}^T(\tau-i) \hat{\vartheta}_{j,k}(\tau), \quad (38)$$

$$\hat{w}_k(\tau-i) = y(\tau-i) - \hat{\phi}_k^T(\tau-i) \hat{\vartheta}_k(\tau), \quad (39)$$

$$\begin{aligned}\bar{\mu}_k(\tau) &\leq \max \left[2\lambda_{\max}^{-1} \sum_{i=\tau-p+1}^{\tau} \hat{\phi}_k(i) \hat{\phi}_k^T(i), \right. \\ &\quad \left. 2\lambda_{\max}^{-1} \sum_{i=\tau-p+1}^{\tau} \hat{\psi}_k(i) \hat{\psi}_k^T(i) \right],\end{aligned}\quad (40)$$

$$\hat{\vartheta}_k(\tau) = \left[\hat{\vartheta}_{1,k}^T(\tau), \hat{\vartheta}_{2,k}^T(\tau), \dots, \hat{\vartheta}_{r,k}^T(\tau) \right]^T, \quad (41)$$

$$\begin{aligned}\hat{\vartheta}_{j,k}(\tau) &= \left[\hat{a}_{j1,k}(\tau), \hat{a}_{j2,k}(\tau), \dots, \hat{a}_{jn_j,k}(\tau), \hat{b}_{j1,k}(\tau), \right. \\ &\quad \left. \hat{b}_{j2,k}(\tau), \dots, \hat{b}_{jn_j,k}(\tau) \right]^T,\end{aligned}\quad (42)$$

$$\hat{\mathbf{c}}_k(\tau) = \left[\hat{c}_{1,k}(\tau), \hat{c}_{2,k}(\tau), \dots, \hat{c}_{n_c,k}(\tau) \right]^T. \quad (43)$$

The steps of computing the parameter estimates $\hat{\vartheta}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$ for the multi-input OEAR systems are as follows.

- (1) Set the data length p , let $\tau = p$, and collect the input-output data $\{u_1(i), u_2(i), \dots, u_r(i), y(i) : i = 0, 1, \dots, p-1\}$.
- (2) Collect the input-output data $\{u_1(\tau), u_2(\tau), \dots, u_r(\tau)\}$ and $y(\tau)$.
- (3) To initialize, let $k = 1$, $\hat{\vartheta}_0(\tau) = \mathbf{1}_{n_0}/p_0$, $\hat{\mathbf{c}}_0(\tau) = \mathbf{1}_{n_c}/p_0$, $\hat{x}_{j,0}(\tau-i) = 1/p_0$, $\hat{w}_0(\tau-i) = 1/p_0$ for $i = 1, 2, \dots, \max[n_j, n_c]$.
- (4) Form $\hat{\phi}_{j,k}(\tau)$ and $\hat{\psi}_k(\tau)$ by (36) and (37), and form $\hat{\phi}_k(\tau)$ by (35).
- (5) Choose a $\bar{\mu}_k(\tau)$ satisfying (40) and update the estimate $\hat{\vartheta}_k(\tau)$ using (33) and $\hat{\mathbf{c}}_k(\tau)$ using (34).
- (6) Read $\hat{\vartheta}_k(\tau)$ and $\hat{\vartheta}_{j,k}(\tau)$ using (41) and (42) and compute $\hat{x}_{j,k}(\tau-i)$ using (38) and $\hat{w}_k(\tau-i)$ using (39).
- (7) Give a small positive $\varepsilon > 0$. If $\|\hat{\vartheta}_k(\tau) - \hat{\vartheta}_{k-1}(\tau)\| + \|\hat{\mathbf{c}}_k(\tau) - \hat{\mathbf{c}}_{k-1}(\tau)\| > \varepsilon$, increase k by 1 and go to Step (4); otherwise, obtain the parameters $\hat{\vartheta}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$ and increase s by 1 and go to Step (2).

5. The Hierarchical Least Squares Based Iterative Algorithm

The H-GI algorithm can produce higher parameter estimation accuracy compared with the GI algorithm, but it

converges slowly. In order to solve this short board, we derive a hierarchical least squares based iterative algorithm for the multi-input OEAR systems.

Minimizing $J_1(\boldsymbol{\theta}, \mathbf{c})$ and letting the partial derivative of $J_1(\boldsymbol{\theta}, \mathbf{c})$ with respect to $\boldsymbol{\theta}$ be zero and minimizing $J_2(\boldsymbol{\theta}, \mathbf{c})$ and letting the partial derivative of $J_2(\boldsymbol{\theta}, \mathbf{c})$ with respect to \mathbf{c} be zero, respectively, we can obtain the least squares estimate $\hat{\boldsymbol{\theta}}(\tau)$:

$$\hat{\boldsymbol{\theta}}(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) \boldsymbol{\phi}^T(i) \right]^{-1} \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) y_1(i) \right], \quad (44)$$

$$\hat{\mathbf{c}}(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) \boldsymbol{\psi}^T(i) \right]^{-1} \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) y_2(i) \right]. \quad (45)$$

Inserting (19) into (44) and (20) into (45) gives

$$\hat{\boldsymbol{\theta}}(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) \boldsymbol{\phi}^T(i) \right]^{-1} \cdot \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\phi}(i) [y(i) - \boldsymbol{\psi}^T(i) \mathbf{c}] \right], \quad (46)$$

$$\hat{\mathbf{c}}(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) \boldsymbol{\psi}^T(i) \right]^{-1} \cdot \left[\sum_{i=\tau-p+1}^{\tau} \boldsymbol{\psi}(i) [y(i) - \boldsymbol{\phi}^T(i) \boldsymbol{\theta}] \right]. \quad (47)$$

The above estimates $\hat{\boldsymbol{\theta}}(\tau)$ and $\hat{\mathbf{c}}(\tau)$ are impossible to compute, since the right-hand side of (46) contains the unknown parameter vector \mathbf{c} and the unknown information vectors $\boldsymbol{\phi}(i)$ and $\boldsymbol{\psi}(i)$ and the right-hand side of (47) also contains the unknown parameter vector $\boldsymbol{\theta}$ and the unknown information vectors $\boldsymbol{\phi}(i)$ and $\boldsymbol{\psi}(i)$. We solve this difficulty by replacing $\boldsymbol{\phi}(i)$, $\boldsymbol{\psi}(i)$ with their estimates $\hat{\boldsymbol{\phi}}_k(i)$, $\hat{\boldsymbol{\psi}}_k(i)$ and replacing \mathbf{c} in (46) and $\boldsymbol{\theta}$ in (47) with their estimates $\hat{\mathbf{c}}_{k-1}(\tau)$ and $\hat{\boldsymbol{\theta}}_k(\tau)$. Then, we can summarize the hierarchical least squares based iterative (LSI) algorithm of estimating the parameter vectors $\boldsymbol{\theta}$ and \mathbf{c} as follows:

$$\hat{\boldsymbol{\theta}}_k(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\phi}}_k(i) \hat{\boldsymbol{\phi}}_k^T(i) \right]^{-1} \cdot \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\phi}}_k(i) [y(i) - \hat{\boldsymbol{\psi}}_k^T(i) \hat{\mathbf{c}}_{k-1}(\tau)] \right], \quad (48)$$

$$\hat{\mathbf{c}}_k(\tau) = \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\psi}}_k(i) \hat{\boldsymbol{\psi}}_k^T(i) \right]^{-1} \cdot \left[\sum_{i=\tau-p+1}^{\tau} \hat{\boldsymbol{\psi}}_k(i) [y(i) - \hat{\boldsymbol{\phi}}_k^T(i) \hat{\boldsymbol{\theta}}_k(\tau)] \right], \quad (49)$$

$$\hat{\boldsymbol{\phi}}_k(\tau) = \left[\hat{\boldsymbol{\phi}}_{1,k}^T(\tau), \hat{\boldsymbol{\phi}}_{2,k}^T(\tau), \dots, \hat{\boldsymbol{\phi}}_{r,k}^T(\tau) \right]^T, \quad (50)$$

$$\hat{\boldsymbol{\phi}}_{j,k}(\tau) = \left[-\hat{x}_{j,k-1}(\tau-1), \dots, -\hat{x}_{j,k-1}(\tau-n_j), u_j(\tau-1), \dots, u_j(\tau-n_j) \right]^T, \quad (51)$$

$$\hat{\boldsymbol{\psi}}_k(\tau) = \left[-\hat{w}_{k-1}(\tau-1), -\hat{w}_{k-1}(\tau-2), \dots, -\hat{w}_{k-1}(\tau-n_c) \right]^T, \quad (52)$$

$$\hat{x}_{j,k}(\tau-i) = \hat{\boldsymbol{\phi}}_{j,k}^T(\tau-i) \hat{\boldsymbol{\theta}}_{j,k}(\tau), \quad (53)$$

$$\hat{w}_k(\tau-i) = y(\tau-i) - \hat{\boldsymbol{\phi}}_k^T(\tau-i) \hat{\boldsymbol{\theta}}_k(\tau), \quad (54)$$

$$\hat{\boldsymbol{\theta}}_k(\tau) = \left[\hat{\boldsymbol{\theta}}_{1,k}^T(\tau), \hat{\boldsymbol{\theta}}_{2,k}^T(\tau), \dots, \hat{\boldsymbol{\theta}}_{r,k}^T(\tau) \right]^T. \quad (55)$$

The procedure for computing the parameter estimation $\hat{\boldsymbol{\theta}}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$ is as follows.

- (1) Give the data length p , let $\tau = p$, collect the input-output data $\{u_1(i), u_2(i), \dots, u_r(i), y(i) : i = 1, \dots, p-1\}$, and give a small positive $\varepsilon > 0$.
- (2) Collect the input-output data $\{u_1(\tau), u_2(\tau), \dots, u_r(\tau)\}$ and $y(\tau)$.
- (3) To initialize, let $k = 1$, $\hat{x}_{j,0}(\tau-i) = 1/p_0$, $\hat{w}_0(\tau-i) = 1/p_0$ for $i = 1, 2, \dots, \max[n_j, n_c]$.
- (4) Form $\hat{\boldsymbol{\phi}}_{j,k}(\tau)$, $\hat{\boldsymbol{\psi}}_k(\tau)$, and $\hat{\boldsymbol{\phi}}_k(\tau)$ by (51), (52), and (50), respectively.
- (5) Update the estimates $\hat{\boldsymbol{\theta}}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$ by (48) and (49) and read $\hat{\boldsymbol{\theta}}_k(\tau)$ by (55).
- (6) Compute $\hat{x}_{j,k}(\tau)$ by (53) and $\hat{w}_k(\tau)$ by (54).
- (7) If $\|\hat{\boldsymbol{\theta}}_k(\tau) - \hat{\boldsymbol{\theta}}_{k-1}(\tau)\| + \|\hat{\mathbf{c}}_k(\tau) - \hat{\mathbf{c}}_{k-1}(\tau)\| > \varepsilon$, increase k by 1 and go to Step (4); otherwise, obtain the parameters $\hat{\boldsymbol{\theta}}_k(\tau)$ and $\hat{\mathbf{c}}_k(\tau)$ and increase s by 1 and go to Step (2).

6. Example

Example 1. Consider the following two-input OEAR system:

$$y(\tau) = \frac{B_1(z)}{A_1(z)} u_1(\tau) + \frac{B_2(z)}{A_2(z)} u_2(\tau) + \frac{1}{C(z)} v(\tau),$$

$$A_1(z) = 1 + a_{11}z^{-1} + a_{12}z^{-2} = 1 + 0.35z^{-1} + 0.27z^{-2},$$

$$B_1(z) = b_{11}z^{-1} + b_{12}z^{-2} = 0.78z^{-1} - 0.40z^{-2},$$

TABLE 1: The LSI parameter estimates and errors for Example 1.

k	a_{11}	a_{12}	b_{11}	b_{12}	a_{21}	a_{22}	b_{21}	b_{22}	c_1	δ (%)
1	-0.00174	0.00053	0.78866	-0.68631	0.00895	-0.00818	0.57027	0.61302	-0.00050	55.79849
2	0.35369	0.27184	0.78727	-0.40443	-0.17629	0.15694	0.55786	0.51059	0.27174	12.15009
3	0.35056	0.26746	0.78457	-0.40524	-0.19397	0.17507	0.56099	0.50495	0.38379	3.57156
4	0.35033	0.26820	0.78447	-0.40537	-0.18622	0.17255	0.56123	0.50724	0.38677	3.52725
5	0.35051	0.26838	0.78449	-0.40522	-0.18616	0.16982	0.56117	0.50718	0.38644	3.58654
6	0.35054	0.26843	0.78450	-0.40520	-0.18807	0.17162	0.56114	0.50618	0.38616	3.53048
7	0.35054	0.26839	0.78449	-0.40520	-0.18776	0.17156	0.56117	0.50634	0.38641	3.52131
8	0.35053	0.26838	0.78449	-0.40521	-0.18765	0.17141	0.56116	0.50639	0.38639	3.52730
9	0.35053	0.26839	0.78449	-0.40520	-0.18771	0.17145	0.56116	0.50636	0.38637	3.52652
10	0.35053	0.26839	0.78449	-0.40520	-0.18771	0.17146	0.56116	0.50636	0.38638	3.52588
True values	0.35000	0.27000	0.78000	-0.40000	-0.20000	0.18000	0.56000	0.50000	0.43000	

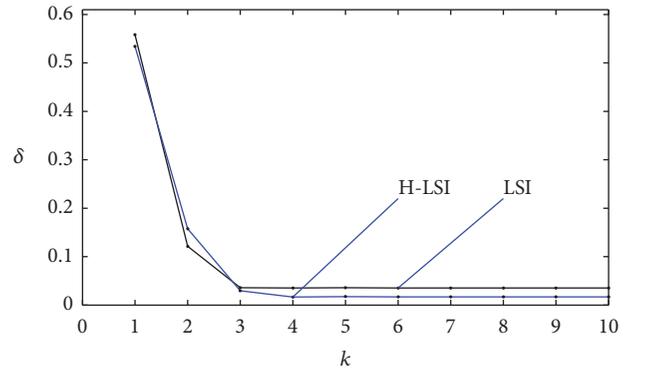
TABLE 2: The H-LSI parameter estimates and errors for Example 1.

k	a_{11}	a_{12}	b_{11}	b_{12}	a_{21}	a_{22}	b_{21}	b_{22}	c_1	δ (%)
1	-0.00165	0.00045	0.78903	-0.68649	0.00000	0.00000	0.57043	0.61368	0.04854	53.42619
2	0.33787	0.25774	0.78598	-0.41664	-0.17740	0.16201	0.56029	0.50972	0.22291	15.77773
3	0.35202	0.26929	0.78458	-0.40359	-0.18973	0.17231	0.56283	0.50476	0.39352	2.96102
4	0.35065	0.26837	0.78458	-0.40503	-0.18892	0.17175	0.56297	0.50464	0.44493	1.66301
5	0.35054	0.26835	0.78458	-0.40512	-0.18923	0.17183	0.56296	0.50444	0.44660	1.73488
6	0.35052	0.26832	0.78458	-0.40514	-0.18926	0.17188	0.56296	0.50442	0.44567	1.68306
7	0.35053	0.26833	0.78457	-0.40514	-0.18926	0.17187	0.56296	0.50443	0.44578	1.68902
8	0.35052	0.26833	0.78457	-0.40514	-0.18926	0.17187	0.56296	0.50443	0.44578	1.68914
9	0.35052	0.26833	0.78457	-0.40514	-0.18926	0.17187	0.56296	0.50443	0.44578	1.68894
10	0.35052	0.26833	0.78457	-0.40514	-0.18926	0.17187	0.56296	0.50443	0.44578	1.68898
True values	0.35000	0.27000	0.78000	-0.40000	-0.20000	0.18000	0.56000	0.50000	0.43000	

$$\begin{aligned}
A_2(z) &= 1 + a_{21}z^{-1} + a_{22}z^{-2} = 1 - 0.20z^{-1} \\
&\quad + 0.18z^{-2}, \\
B_2(z) &= b_{21}z^{-1} + b_{22}z^{-2} = 0.56z^{-1} + 0.50z^{-2}, \\
C(z) &= 1 + c_1z^{-1} = 1 + 0.43z^{-1}, \\
\boldsymbol{\theta} &= [a_{11}, a_{12}, b_{11}, b_{12}, a_{21}, a_{22}, b_{21}, b_{22}, c_1]^T \\
&= [0.35, 0.27, 0.78, -0.40, \\
&\quad -0.20, 0.18, 0.56, 0.50, 0.43]^T.
\end{aligned} \tag{56}$$

The inputs $\{u_1(\tau), u_2(\tau)\}$ are taken as two persistent excitation signal sequences with zero mean and unit variance and $\{v(\tau)\}$ as a white noise sequence with zero mean and variance $\sigma^2 = 0.20^2$.

Take the data length $L = 2000$, applying the LSI algorithm and the H-LSI algorithm to estimate the parameters of this example system. The parameter estimates and their errors of the LSI algorithm are shown in Table 1, the parameter estimates and their errors of H-LSI algorithm are shown in Table 2, and the parameter estimation errors of the LSI and H-LSI algorithms versus k are shown in Figure 1.

FIGURE 1: The GI estimation errors δ versus k with different σ^2 .

From the simulation results in Tables 1 and 2 and Figure 1, we can draw the following conclusions.

- (i) The estimation errors given by the LSI algorithm and H-LSI algorithm become smaller and smaller as iteration variable k increases.
- (ii) Under the same noise variance, the estimation errors given by the H-LSI algorithm are lower than that given by the LSI algorithm.

TABLE 3: The H-LSI parameter estimates and errors for Example 2.

k	a_{11}	a_{12}	b_{11}	b_{12}	a_{21}	a_{22}	b_{21}	b_{22}	c_1	δ (%)
1	0.00000	0.00000	0.55541	-0.78641	0.05229	-0.02811	0.59789	-0.92588	0.01269	34.42173
2	0.14148	-0.30660	0.55273	-0.70377	0.18346	-0.14142	0.60426	-0.82256	0.58395	21.10705
3	0.19132	-0.23272	0.55125	-0.67866	0.21589	-0.09534	0.60251	-0.80142	0.01985	19.82911
4	0.18954	-0.24197	0.55162	-0.67979	0.21649	-0.09730	0.60299	-0.80191	0.37416	5.32335
5	0.19059	-0.23897	0.55155	-0.67915	0.21666	-0.09655	0.60291	-0.80167	0.27810	1.93353
6	0.19032	-0.23977	0.55157	-0.67931	0.21662	-0.09670	0.60293	-0.80172	0.30098	1.11341
7	0.19039	-0.23956	0.55157	-0.67927	0.21663	-0.09667	0.60293	-0.80171	0.29514	1.17583
8	0.19037	-0.23961	0.55157	-0.67928	0.21662	-0.09668	0.60293	-0.80171	0.29664	1.14596
9	0.19038	-0.23960	0.55157	-0.67928	0.21662	-0.09667	0.60293	-0.80171	0.29625	1.15289
10	0.19038	-0.23960	0.55157	-0.67928	0.21662	-0.09668	0.60293	-0.80171	0.29635	1.15101
True values	0.18000	-0.25000	0.55000	-0.68000	0.22000	-0.10000	0.60000	-0.80000	0.30000	

TABLE 4: The H-GI parameter estimates and errors for Example 2.

k	a_{11}	a_{12}	b_{11}	b_{12}	a_{21}	a_{22}	b_{21}	b_{22}	c_1	δ (%)
1	0.06363	-0.03646	0.51672	-0.72852	0.01182	-0.03937	0.54579	-0.85581	0.00627	31.71160
2	0.17357	-0.19236	0.53839	-0.74274	0.14494	-0.16525	0.57604	-0.89230	0.09049	18.71188
5	0.18370	-0.23560	0.55059	-0.69456	0.15382	-0.14339	0.59693	-0.85356	0.09666	15.92395
10	0.18597	-0.24230	0.55243	-0.68240	0.17824	-0.12609	0.60144	-0.83140	0.10937	14.08831
20	0.18821	-0.24117	0.55225	-0.68049	0.20204	-0.10795	0.60193	-0.81289	0.13405	11.85943
50	0.18994	-0.23981	0.55225	-0.67953	0.21588	-0.09733	0.60184	-0.80231	0.18766	8.00236
True values	0.18000	-0.25000	0.55000	-0.68000	0.22000	-0.10000	0.60000	-0.80000	0.30000	

TABLE 5: The H-GI parameter estimates and errors for Example 2.

k	a_{11}	a_{12}	b_{11}	b_{12}	a_{21}	a_{22}	b_{21}	b_{22}	c_1	δ (%)
1	0.02264	-0.01297	0.18381	-0.25916	0.02264	-0.01297	0.19416	-0.30444	0.00295	68.31458
2	0.09666	-0.06071	0.34760	-0.48848	0.10932	-0.05447	0.36923	-0.57459	0.28748	34.49578
5	0.24231	-0.19630	0.53350	-0.64647	0.22618	-0.09991	0.57665	-0.77609	0.33914	7.35051
10	0.23004	-0.20855	0.55156	-0.65412	0.22289	-0.09268	0.59940	-0.79531	0.33141	5.44993
20	0.21210	-0.22233	0.55260	-0.66565	0.21857	-0.09528	0.60089	-0.80002	0.31890	3.45097
50	0.19366	-0.23682	0.55262	-0.67729	0.21663	-0.09677	0.60117	-0.80156	0.29411	1.47162
True values	0.18000	-0.25000	0.55000	-0.68000	0.22000	-0.10000	0.60000	-0.80000	0.30000	

(iii) The estimation accuracy of the H-LSI algorithm is close to their true values; this indicates that the proposed algorithm can effectively identify the multi-input OEAR systems.

Example 2. Consider the following another two-input OEAR system:

$$y(\tau) = \frac{B_1(z)}{A_1(z)}u_1(\tau) + \frac{B_2(z)}{A_2(z)}u_2(\tau) + \frac{1}{C(z)}v(\tau),$$

$$A_1(z) = 1 + a_{11}z^{-1} + a_{12}z^{-2} = 1 + 0.30z^{-1} - 0.20z^{-2},$$

$$B_1(z) = b_{11}z^{-1} + b_{12}z^{-2} = 0.55z^{-1} - 0.80z^{-2},$$

$$A_2(z) = 1 + a_{21}z^{-1} + a_{22}z^{-2} = 1 + 0.20z^{-1} - 0.10z^{-2},$$

$$B_2(z) = b_{21}z^{-1} + b_{22}z^{-2} = 0.40z^{-1} - 0.90z^{-2},$$

$$C(z) = 1 + c_1z^{-1} = 1 + 0.41z^{-1},$$

$$\theta = [a_{11}, a_{12}, b_{11}, b_{12}, a_{21}, a_{22}, b_{21}, b_{22}, c_1]^T = [0.30,$$

$$-0.20, 0.55, -0.80, 0.20, -0.10, 0.40,$$

$$-0.90, 0.41]^T.$$

(57)

The simulation conditions are the same as that of Example 1, and the noise variance $\sigma^2 = 0.20^2$. Take the data length $L = 3000$. Applying the GI algorithm and the H-GI algorithm to estimate the parameters of this example system, the simulation results are shown in Tables 3–5 and Figure 2.

From the simulation results in Tables 3–5 and Figure 2, we can draw the following conclusions.

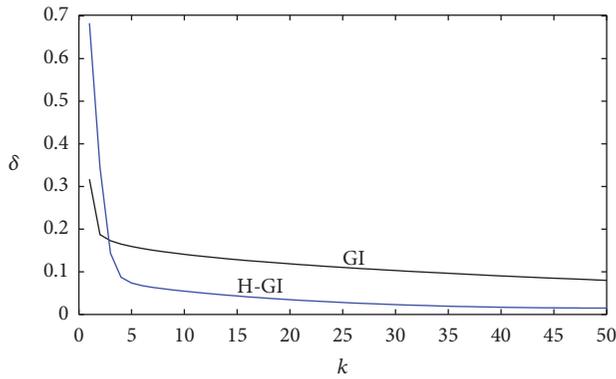


FIGURE 2: The GI and H-GI estimation errors δ versus k .

- (i) Under the same noise variance and data length, the H-GI algorithm has less estimation errors than the GI algorithm. This shows that the H-GI estimation algorithm can obtain more accurate estimates than the GI algorithm.
- (ii) As the iteration variable k increases, the H-GI parameter estimates are very close to their true values.
- (iii) The proposed H-GI algorithm requires more iterations than the H-LSI algorithm to achieve almost same estimation accuracy.

7. Conclusions

Combining the iterative technique and the hierarchical identification principle, a H-GI algorithm and a H-LSI algorithm are derived for identifying the multi-input OEAR systems. Compared with the GI algorithm, the H-GI algorithm can generate more accurate parameter estimates. Compared with the H-GI algorithm, the H-LSI algorithm has faster convergence speed. The proposed methods can be extended to discuss the parameter estimation of the multi-input-output systems with colored noise [38–42] and time-delay systems [43, 44], such as network and signal processing [45–52].

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The author is grateful to her supervisor Professor Feng Ding at the Jiangnan University for his helpful suggestions and the main idea of this work comes from him and his book *Multi-Innovation Identification Theory and Methods*, Beijing: Science Press, 2016. This work was supported by the Natural Science Foundation of Shandong Province (China, ZR2016FL08) and the Science Foundation of Jining University (China, 2016QNKJ01).

References

- [1] L. Xu, "A proportional differential control method for a time-delay system using the Taylor expansion approximation," *Applied Mathematics and Computation*, vol. 236, pp. 391–399, 2014.
- [2] L. Xu, L. Chen, and W. Xiong, "Parameter estimation and controller design for dynamic systems from the step responses based on the Newton iteration," *Nonlinear Dynamics*, vol. 79, no. 3, pp. 2155–2163, 2015.
- [3] K. Rajarathinam, J. B. Gomm, D.-L. Yu, and A. S. Abdelhadi, "PID controller tuning for a multivariable glass furnace process by genetic algorithm," *International Journal of Automation and Computing*, vol. 13, no. 1, pp. 64–72, 2016.
- [4] D. Zumoffen, M. Gonzalo, and M. Basualdo, "Improvements on multivariable control strategies tested on the Petlyuk distillation column," *Chemical Engineering Science*, vol. 93, pp. 292–306, 2013.
- [5] J. Na, M. N. Mahyuddin, G. Herrmann, X. Ren, and P. Barber, "Robust adaptive finite-time parameter estimation and control for robotic systems," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 16, pp. 3045–3071, 2015.
- [6] H. Li, Y. Shi, and W. Yan, "On Neighbor Information Utilization in Distributed Receding Horizon Control for Consensus-Seeking," *IEEE Transactions on Cybernetics*, vol. 46, no. 9, pp. 2019–2027, 2016.
- [7] L. Feng, M. Wu, Q. Li et al., "Array Factor Forming for Image Reconstruction of One-Dimensional Nonuniform Aperture Synthesis Radiometers," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 2, pp. 237–241, 2016.
- [8] G. Gu, S. Wan, and L. Qiu, "Networked stabilization for multi-input systems over quantized fading channels," *Automatica*, vol. 61, pp. 1–8, 2015.
- [9] A. Cristofaro and S. Pettinari, "Fault accommodation for multi-input linear sampled-data systems," *International Journal of Adaptive Control and Signal Processing*, vol. 29, no. 7, pp. 835–854, 2015.
- [10] Y. Ji and X. Liu, "Unified synchronization criteria for hybrid switching-impulsive dynamical networks," *Circuits, Systems and Signal Processing*, vol. 34, no. 5, pp. 1499–1517, 2015.
- [11] L. Jia, X. Li, and M.-S. Chiu, "Correlation analysis based MIMO neuro-fuzzy Hammerstein model with noises," *Journal of Process Control*, vol. 41, pp. 76–91, 2016.
- [12] F. Ding, F. Wang, L. Xu, and M. Wu, "Decomposition based least squares iterative identification algorithm for multivariate pseudo-linear ARMA systems using the data filtering," *Journal of The Franklin Institute*, vol. 354, no. 3, pp. 1321–1339, 2017.
- [13] C. Yu and M. Verhaegen, "Blind multivariable ARMA subspace identification," *Automatica*, vol. 66, pp. 3–14, 2016.
- [14] G. Mercère and L. Bako, "Parameterization and identification of multivariable state-space systems: a canonical approach," *Automatica*, vol. 47, no. 8, pp. 1547–1555, 2011.
- [15] M. Li, X. Liu, and F. Ding, "Least-squares-based iterative and gradient-based iterative estimation algorithms for bilinear systems," *Nonlinear Dynamics*, vol. 89, no. 1, pp. 197–211, 2017.
- [16] M. Li, X. Liu, and F. Ding, "The maximum likelihood least squares based iterative estimation algorithm for bilinear systems with autoregressive moving average noise," *Journal of The Franklin Institute*, vol. 354, no. 12, pp. 4861–4881, 2017.
- [17] Y. Wang and F. Ding, "Recursive least squares algorithm and gradient algorithm for Hammerstein-Wiener systems using the

- data filtering,” *Nonlinear Dynamics*, vol. 84, no. 2, pp. 1045–1053, 2016.
- [18] R. C. Panda and S. Vijayaraghavan, “Parameter estimation of linear MIMO systems using sequential relay feedback test,” *AIChE Journal*, vol. 60, no. 5, pp. 1672–1681, 2014.
- [19] M. Jafari, M. Salimifard, and M. Dehghani, “Identification of multivariable nonlinear systems in the presence of colored noises using iterative hierarchical least squares algorithm,” *ISA Transactions*, vol. 53, no. 4, pp. 1243–1252, 2014.
- [20] Y. Wang and F. Ding, “Novel data filtering based parameter identification for multiple-input multiple-output systems using the auxiliary model,” *Automatica*, vol. 71, pp. 308–313, 2016.
- [21] J. L. Ding, “Recursive and iterative least squares parameter estimation algorithms for multiple-input-output-error systems with autoregressive noise,” *Circuits, Systems and Signal Processing*.
- [22] H.-N. Wu and J.-W. Wang, “Observer design and output feedback stabilization for nonlinear multivariable systems with diffusion PDE-governed sensor dynamics,” *Nonlinear Dynamics*, vol. 72, no. 3, pp. 615–628, 2013.
- [23] X. Wang and F. Ding, “Recursive parameter and state estimation for an input nonlinear state space system using the hierarchical identification principle,” *Signal Processing*, vol. 117, pp. 208–218, 2015.
- [24] F. Ding, F. Wang, L. Xu, T. Hayat, and A. Alsaedi, “Parameter estimation for pseudo-linear systems using the auxiliary model and the decomposition technique,” *IET Control Theory & Applications*, vol. 11, no. 3, pp. 390–400, 2017.
- [25] C. Schranz, C. Knöbel, J. Kretschmer, Z. Zhao, and K. Möller, “Hierarchical parameter identification in models of respiratory mechanics,” *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 11, pp. 3234–3241, 2011.
- [26] L. Xu and F. Ding, “Parameter estimation algorithms for dynamical response signals based on the multi-innovation theory and the hierarchical principle,” *IET Signal Processing*, vol. 11, no. 2, pp. 228–237, 2017.
- [27] J. Na, J. Yang, X. Wu, and Y. Guo, “Robust adaptive parameter estimation of sinusoidal signals,” *Automatica*, vol. 53, pp. 376–384, 2015.
- [28] L. Xu, “The damping iterative parameter identification method for dynamical systems based on the sine signal measurement,” *Signal Processing*, vol. 120, pp. 660–667, 2016.
- [29] L. Xu, “Application of the Newton iteration algorithm to the parameter estimation for dynamical systems,” *Journal of Computational and Applied Mathematics*, vol. 288, pp. 33–43, 2015.
- [30] F. Ding, L. Xu, and Q. Zhu, “Performance analysis of the generalised projection identification for time-varying systems,” *IET Control Theory & Applications*, vol. 10, no. 18, pp. 2506–2514, 2016.
- [31] M. Li, X. Liu, and F. Ding, “The Gradient-Based Iterative Estimation Algorithms for BILinear Systems with AUToregressive Noise,” *Circuits, Systems and Signal Processing*, vol. 36, no. 11, pp. 4541–4568, 2017.
- [32] D. Meng, “Recursive least squares and multi-innovation gradient estimation algorithms for bilinear stochastic systems,” *Circuits, Systems and Signal Processing*, vol. 36, no. 3, pp. 1052–1065, 2017.
- [33] L. Xu and F. Ding, “Recursive Least Squares and Multi-innovation Stochastic Gradient Parameter Estimation Methods for Signal Modeling,” *Circuits, Systems and Signal Processing*, vol. 36, no. 4, pp. 1735–1753, 2017.
- [34] X. Liu and J. Lu, “Least squares based iterative identification for a class of multirate systems,” *Automatica*, vol. 46, no. 3, pp. 549–554, 2010.
- [35] X. Wan, Y. Li, C. Xia, M. Wu, J. Liang, and N. Wang, “A T-wave alternans assessment method based on least squares curve fitting technique,” *Measurement*, vol. 86, pp. 93–100, 2016.
- [36] F. Ding, X. M. Liu, H. B. Chen, and G. Y. Yao, “Hierarchical gradient based and hierarchical least squares based iterative parameter identification for CARARMA systems,” *Signal Processing*, vol. 97, pp. 31–39, 2014.
- [37] J. Ma, W. Xiong, and F. Ding, “Iterative identification algorithms for input nonlinear output error autoregressive systems,” *International Journal of Control, Automation, and Systems*, vol. 14, no. 1, pp. 140–147, 2016.
- [38] D. Wang, L. Mao, and F. Ding, “Recasted models-based hierarchical extended stochastic gradient method for MIMO nonlinear systems,” *IET Control Theory & Applications*, vol. 11, no. 4, pp. 476–485, 2017.
- [39] D.-Q. Wang, Z. Zhang, and J.-Y. Yuan, “Maximum likelihood estimation method for dual-rate Hammerstein systems,” *International Journal of Control, Automation, and Systems*, vol. 15, no. 2, pp. 698–705, 2017.
- [40] D.-Q. Wang, H.-B. Liu, and F. Ding, “Highly efficient identification methods for dual-rate hammerstein systems,” *IEEE Transactions on Control Systems Technology*, vol. 23, no. 5, pp. 1952–1960, 2015.
- [41] D. Wang and F. Ding, “Parameter estimation algorithms for multivariable Hammerstein CARMA systems,” *Information Sciences*, vol. 355–356, pp. 237–248, 2016.
- [42] D. Wang, “Hierarchical parameter estimation for a class of MIMO Hammerstein systems based on the reframed models,” *Applied Mathematics Letters*, vol. 57, pp. 13–19, 2016.
- [43] F. Ding, X. Wang, L. Mao, and L. Xu, “Joint state and multi-innovation parameter estimation for time-delay linear systems and its convergence based on the Kalman filtering,” *Digital Signal Processing*, vol. 62, pp. 211–223, 2017.
- [44] L. Xu, F. Ding, Y. Gu, A. Alsaedi, and T. Hayat, “A multi-innovation state and parameter estimation algorithm for a state space system with d-step state-delay,” *Signal Processing*, vol. 140, pp. 97–103, 2017.
- [45] J. Na, J. Yang, X. Ren, and Y. Guo, “Robust adaptive estimation of nonlinear system with time-varying parameters,” *International Journal of Adaptive Control and Signal Processing*, vol. 29, no. 8, pp. 1055–1072, 2015.
- [46] J. Pan, X. Yang, H. Cai, and B. Mu, “Image noise smoothing using a modified Kalman filter,” *Neurocomputing*, vol. 173, pp. 1625–1629, 2016.
- [47] J. Pan, X. Jiang, X. Wan, and W. Ding, “A filtering based multi-innovation extended stochastic gradient algorithm for multivariable control systems,” *International Journal of Control, Automation, and Systems*, vol. 15, no. 3, pp. 1189–1197, 2017.
- [48] N. Zhao, M. Wu, and J. Chen, “Android-based mobile educational platform for speech signal processing,” *International Journal of Electrical Engineering Education*, vol. 54, no. 1, pp. 3–16, 2017.
- [49] X. F. Li, Y. D. Chu, A. Y. T. Leung, and H. Zhang, “Synchronization of uncertain chaotic systems via complete-adaptive-impulsive controls,” *Chaos Solitons Fractals*, vol. 100, pp. 24–30, 2017.
- [50] Y. Ji and F. Ding, “Multiperiodicity and exponential attractivity of neural networks with mixed delays,” *Circuits, Systems and Signal Processing*, vol. 36, no. 6, pp. 2558–2573, 2017.

- [51] N. Zhao, Y. Chen, R. Liu, M. H. Wu, and W. Xiong, "Monitoring strategy for relay incentive mechanism in cooperative communication networks," *Computers & Electrical Engineering*, vol. 60, pp. 14–29, 2017.
- [52] Y. Wang, F. Ding, and L. Xu, "Some new results of designing an IIR filter with colored noise for signal processing," *Digital Signal Processing*, vol. 72, pp. 44–58, 2018.

Research Article

Applying Two-Stage Neural Network Based Classifiers to the Identification of Mixture Control Chart Patterns for an SPC-EPC Process

Yuehjen E. Shao,¹ Po-Yu Chang,¹ and Chi-Jie Lu²

¹Department of Statistics and Information Science, Fu Jen Catholic University, New Taipei City, Taiwan

²Department of Industrial Management, Chien Hsin University of Science and Technology, Zhongli, Taoyuan County 32097, Taiwan

Correspondence should be addressed to Yuehjen E. Shao; stat1003@mail.fju.edu.tw

Received 7 July 2017; Revised 6 September 2017; Accepted 19 September 2017; Published 22 October 2017

Academic Editor: Yanan Li

Copyright © 2017 Yuehjen E. Shao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The effective controlling and monitoring of an industrial process through the integration of statistical process control (SPC) and engineering process control (EPC) has been widely addressed in recent years. However, because the mixture types of disturbances are often embedded in underlying processes, mixture control chart patterns (MCCPs) are very difficult for an SPC-EPC process to identify. This can result in problems when attempting to determine the underlying root causes of process faults. Additionally, a large number of categories of disturbances may be present in a process, but typical single-stage classifiers have difficulty in identifying large numbers of categories of disturbances in an SPC-EPC process. Therefore, we propose a two-stage neural network (NN) based scheme to enhance the accurate identification rate (AIR) for MCCPs by performing dimension reduction on disturbance categories. The two-stage scheme includes a combination of a NN, support vector machine (SVM), and multivariate adaptive regression splines (MARS). Experimental results reveal that the proposed scheme achieves a satisfactory AIR for identifying MCCPs in an SPC-EPC system.

1. Introduction

1.1. Background. High-quality products are typically manufactured through stable and effective industrial processes. Stable and effective processes are often complex, meaning that they must be well controlled and monitored using various advanced techniques. Although it is not new to industry, statistical process control (SPC) is still a widely used quality technique throughout industries. However, difficulty may be encountered when typical SPC charts are used to monitor an autocorrelated process. When the process measurements are autocorrelated, the false alarms are increased, and these improper signals can result in a misinterpretation. The use of engineering process control (EPC) has been successfully proposed to overcome this difficulty [1–4]. Accordingly, the SPC-EPC system plays an important role in industry. From a quality and control engineering viewpoint, SPC and EPC are two very important techniques for maintaining the quality of an industrial process. The main function of SPC is to

monitor a process by plotting an observed sample on SPC control charts [5]. An out-of-control signal will be triggered when the process is unstable. The primary function of EPC is to compensate for the effects of process disturbances by manipulating controllable variables [1, 2]. Although they have their own individual features and specialties, SPC and EPC are naturally linked due to their roles in monitoring and controlling industrial processes. As a result, the goal-oriented integration of these techniques has been studied extensively in recent decades [1–4].

In an integrated mechanism, the role of SPC is to trigger an out-of-control signal when disturbances occur in a process and the role of EPC is to compensate for the effects of these disturbances. During this period, process personnel must identify the root causes of a disturbance and quickly fix them. However, the identification of root causes is difficult for an SPC-EPC process. The major reason for this is that although EPC can compensate for the effects of disturbances, it may conceal the mixture patterns of underlying disturbances.

Because the mixture control chart patterns (MCCPs) are hidden, it is much more difficult to identify the MCCPs in a complex system [6]. Additionally, because different mixture disturbances are typically associated with specific root causes that adversely affect the process, the effective identification of MCCPs for an SPC-EPC process is crucial [7]. Therefore, the issue of identification of MCCPs for industrial processes is an important research topic. Previous research has largely focused on the issue of identification of MCCPs for an SPC application alone. Therefore, the disturbances are assumed to be eliminated immediately after an SPC signal is triggered. In practical applications, the immediate elimination of disturbances is very unlikely. The identification of root causes of disturbances in industrial processes is complex and time-consuming. Consequently, this study mainly focuses on the identification of MCCPs when mixture disturbances exist in an SPC-EPC process.

1.2. Related Work. Due to its importance in the context of process improvement, many studies have investigated control chart pattern (CCP) recognition for industrial processes. For example, some have employed various NNs to recognize CCPs for processes [8, 9]. A NN ensemble-enabled model was used to classify unnatural CCPs for an autocorrelated process [10]. Process observations are assumed to follow an autoregressive with order 1 (i.e., AR(1)) process with an unknown constant coefficient. Because the autocorrelated process observations greatly affect the performance of NN based CCP classifiers, a learning vector quantization NN based approach was proposed to identify CCPs in an industrial process with various levels of autocorrelation [11]. Another team developed a two-module system for identifying six classes of CCPs [12]. The first module extracts features from the data and the second module contains several multilayer perceptron NNs with different parameters and training algorithms. Another group proposed a hybrid system that uses feature extraction modules, classifiers, and optimization modules to recognize common types of CCPs [13]. A multilayer perceptron NN and SVM were employed as classifiers.

In addition to using NN based classifiers to perform CCP recognition, SVM based classifiers are an effective alternative to identifying CCPs for a process. A multistage mechanism that combines independent component analysis and an SVM was proposed for classifying CCPs in a process in [14]. In addition to classifying CCPs for a process, one scheme was also reported to be capable of estimating abnormal pattern parameters by using an SVM and a probabilistic NN [15]. Because a typical SVM classifier may have poor performance, a weighted support vector machine (WSVM) approach was studied for classifying CCPs in an automated process in [16]. The research evaluated the performance of the WSVM approach on a real application from the wafer industry.

Several hybrid approaches have also been proposed for identifying CCPs in processes. A hybrid mechanism was used to combine wavelet filtering and adaptive resonance theory for the recognition of CCPs in a process in [17]. A hybrid approach combining extreme-point symmetric mode decomposition and an extreme learning machine (ELM) was

studied in order to classify concurrent CCPs and accurately quantify major CCP parameters of the specific basic CCPs involved in [18]. A hybrid learning-based model was used to identify various types of unnatural CCPs in a process in [19]. A hybrid feature-based CCP classifier containing classification and regression trees, as well as NN techniques, was proposed for recognizing CCPs in [20].

In [21], it was reported that most related research has focused on single CCP recognition. Therefore, the research developed a hybrid methodology combining wavelet transforms and a particle swarm optimization-based support vector machine for concurrent CCP recognition. Because the assumption of a single type of unnatural pattern exists for the process data, poor CCP recognition may occur when unnatural concurrent CCPs are present in the process data. An approach combining singular spectrum analysis and a learning vector quantization network was proposed in order to recognize concurrent CCPs in [22]. Although this approach is useful for recognizing concurrent CCPs in univariate manufacturing processes, it did not account for the effects of using EPC on the process data. A thorough survey study regarding CCP recognition was performed in [23]. It was reported that a great number of CCP studies between 1991 and 2010 investigated various objectives and conditions. One can refer to [23] for additional details regarding related work on CCP recognition.

In spite of the considerable number of studies discussing the topic of CCPs, relatively little research has focused on methods of identifying a combination of two single CCPs or MCCPs in an SPC-EPC process. Although some have claimed that their approaches are suitable for CCP recognition in manufacturing process data [14, 22], they did not address the effects of EPC on the process data. In [6], although CCP recognition for an SPC-EPC system was investigated, the number of categories of CCPs was relatively small [6]. Additionally, the structure of the CCPs in [6] is of a consecutive type, meaning that the CCPs are easily recognized by one-stage classifiers. Therefore, this study aims to identify MCCPs in an SPC-EPC process. A first-order integrated moving average (IMA(1, 1)) component is assumed to be the noise in the process [1, 4]. Because minimum mean squared error (MMSE) has been successfully used in SPC-EPC processes, an MMSE is employed as the EPC component for the process [1, 4]. In this study, we arbitrarily assume that three individual disturbances can be present in the process simultaneously. Additionally, we focus on identifying the two single CCPs for a process. The reason for not considering single CCPs is that most studies have performed already identification tasks for such situations. The reason for not considering three single (i.e., whole single) CCPs is that we believe that the chance for having a whole single MCCP is very low.

A large number of categories (i.e., more than five) increase the degree of difficulty and lower the accurate identification rate (AIR) for the classification task. For example, in the case of five single disturbances in a process, the team in [6] reported that the values of AIR were 40.00%, 57.73%, 34.84%, and 54.40% with the use of NN, ELM, rough set, and random forest methods, respectively. In this study, because we assume that three single disturbances are present in a

process simultaneously, we have seven categories that must be classified. It is difficult to correctly classify seven categories using typical single-stage classifiers. Therefore, we propose a two-stage identification approach to overcome the problems associated with a large number of categories. Because a NN has data-driven and self-adaptive properties, as well as the ability to capture the nonlinear and complex underlying characteristics in an industrial process with a high degree of accuracy, we employ an NN classifier as the main component in the proposed two-stage mechanism. We also consider a support vector machine (SVM) and multivariate adaptive regression splines (MARS) as components for the proposed mechanism. The reason for choosing an SVM is that, other than ANNs, SVMs are one of the most widely used classifiers for CCP identification. The reason for choosing MARS is that they have not been adopted for CCP identification previously, although MARS are effective at classification and forecasting [24].

Experimental results reveal that the proposed two-stage NN based approach is able to effectively identify various MCCPs in an SPC-EPC process. The remainder of this paper is organized as follows. Section 2 discusses the structure of an industrial process and five types of single disturbances. The difficulty of MCCP identification in an SPC-EPC process is also addressed. The three soft computing techniques used for MCCP identification in this study are introduced in Section 3. Section 4 presents the results of simulation experiments to demonstrate the performance of the proposed approaches. The final section discusses the research findings and conclusions derived from this study.

2. The Process and Disturbance Models

2.1. The Industrial Process Models. A typical SPC process can be expressed as follows:

$$Y_t = u + a_t, \quad (1)$$

where Y_t is process output at time t and u is process mean level; without loss of generality, this study assumes that $u = 0$; a_t is white noise at time t ; the white noise follows a normal probability distribution with mean of 0 and constant variance of σ^2 . Without loss of generality, this study assumes that $\sigma^2 = 1$.

Because autocorrelation widely exists in practical chemical or continuous processes [10, 11, 25–27], the autocorrelation structure should be included in the process model. Therefore, the EPC is usually employed to compensate for effects of the autocorrelation and disturbances. A typical SPC-EPC process can be represented by a well-known zero-order process with the IMA(1, 1) noise [1, 4, 6]; that is,

$$\begin{aligned} Y_{t+1} &= qX_t + d_{t+1}, \\ d_{t+1} &= \frac{(1 - \theta B) a_t}{(1 - B)}, \end{aligned} \quad (2)$$

where d_{t+1} is process noise at time $t + 1$, which follows an IMA(1, 1) process, θ is the parameter of an IMA(1, 1) process, X_t is control variable's measurement at time t , q is the system

gain, which is a certain parameter, and B is backward shift operator, which is defined as $Y_t B^j = Y_{t-j}$ for $j = 1, 2, 3, \dots$

A suitable EPC or MMSE can be obtained when the process model is represented by (2), and it follows that [1]

$$X_t = -\frac{(1 - \theta)}{q} \sum_{j=-\infty}^t Y_j, \quad (3)$$

2.2. The Disturbance Models. Disturbances may upset the process at any time. When a certain disturbance has occurred, the process model in (2) should be reformulated as

$$Y_{t+1} = qX_t + d_{t+1} + D_{t+1}, \quad (4)$$

where D_{t+1} is a certain disturbance at time $t + 1$.

This study considers five single disturbances in an SPC-EPC process, and they are described as follows [28, 29]:

$$\begin{aligned} \text{Cycle (CYC): } D_{t+1} &= D_t^{\text{CYC}} = \sin\left(\frac{2\pi t}{\psi}\right) U_t + a_t \\ \text{Systematic (SYS): } D_{t+1} &= D_t^{\text{SYS}} = g \times (-1)^t + a_t \\ \text{Shift (SHI): } D_{t+1} &= D_t^{\text{SHI}} = D_t + a_t \\ \text{Stratification (STR): } D_{t+1} &= D_t^{\text{STA}} = r\sigma^2 \\ \text{Trend (TRE): } D_{t+1} &= D_t^{\text{TRE}} = tS_t + a_t, \end{aligned} \quad (5)$$

where D_t^{CYC} is cycle disturbance value at time t , U_t is cycle amplitude, which is assumed to follow a uniform distribution within the range of (1.5, 2.5), ψ is cycle period, which is assumed to be $\psi = 8$. D_t^{SYS} is systematic disturbance value at time t , g is magnitude of the systematic pattern in terms of σ^2 , which is assumed to follow a uniform distribution within the range of (1.0, 3.0), D_t^{SHI} is shift disturbance value at time t , D_t is level of shift disturbance, which is assumed to be $D_t = 3$ after shifting, D_t^{STA} is stratification disturbance value at time t , r is random noise, which is assumed to follow a uniform distribution within the range of (0.2, 0.4), D_t^{TRE} is trend disturbance value at time t , and S_t is trend slope, which is assumed to follow a uniform distribution within the range of (0.05, 0.1).

3. The Problems and the Classifiers

3.1. The Problems. When one of those five disturbances is presented in the process, we refer to the presence of a single CCP. When any two of those five disturbances are concurrently presented in the process, we refer to the presence of MCCPs in this study. Figure 1 displays the process outputs when a single SHI disturbance has occurred after observation number 50 for an SPC process (i.e., (1)). Figure 2 shows the process outputs when a single SHI disturbance has occurred after observation number 50 for an SPC-EPC process (i.e., (2)). Because an MMSE or EPC is used for compensating for the disturbance's effects, we can observe that the pattern in Figure 2 is different from the pattern in

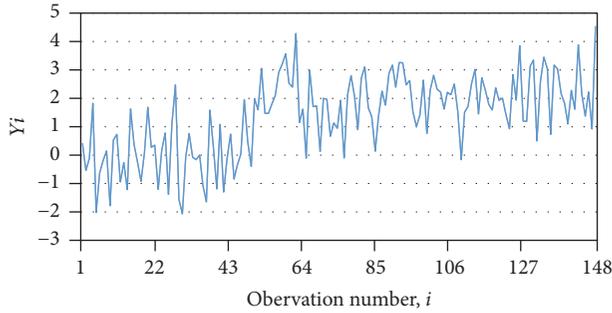


FIGURE 1: The SPC process outputs with the presence of a SHI disturbance intruding after observation number 50.

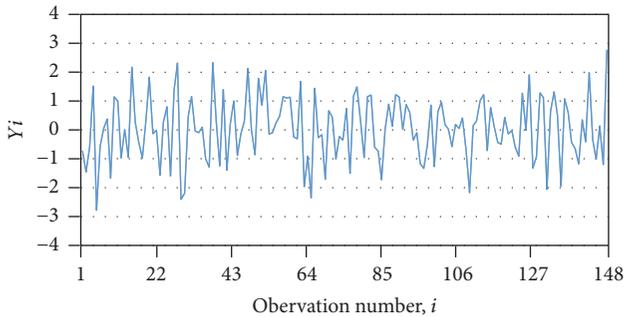


FIGURE 2: The SPC-EPC process outputs with the presence of a SHI disturbance intruding after observation number 50, using the EPC adjustments.

Figure 1. Also, it is easier to identify the CCP in Figure 1, and most of the studies were concerned with the setting in Figure 1 (i.e., SPC alone). In contrast to the setting in Figure 1, this study focuses on the more complex setting in Figure 2 (i.e., an SPC-EPC process). The similar case would happen in other single disturbances as well.

When a single SHI disturbance and a single CYC disturbance concurrently occur, Figures 3 and 4 show the MCCPs for an SPC process and an SPC-EPC process, respectively. When a typical SPC chart is used to monitor the SPC-EPC system, an out-of-control signal will be triggered after observation number 51 in Figure 4. The process personnel can start searching for the root causes by investigating the CCPs. In addition, we can employ exponentially weighted moving average (EWMA) or cumulative sum (CUSUM) control charts to detect smaller magnitude of the disturbances [30, 31]. We can apply Shewhart type of control charts to the detection of a larger magnitude of the disturbances [5]. In addition, by comparison with CCPs in Figure 1 and Figure 3 or 4 for a process, we can notice that the MCCPs in Figure 3 or 4 is more complex to be recognized. By comparison with MCCPs in Figure 3 and Figure 4, the MCCPs in Figure 4 seem to be more difficult to be correctly recognized. Accordingly, this study aims to distinguish each individual single disturbance type for the MCCPs. One can obviously notice that the recognition of MCCPs in Figure 4 is a very difficult task.

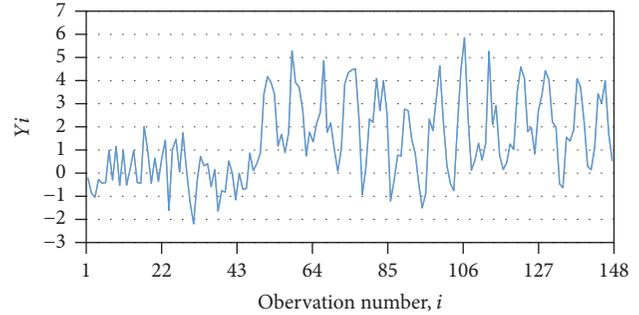


FIGURE 3: The SPC process outputs with the presence of a mixture type of SHI and CYC disturbances intruding after observation number 50.

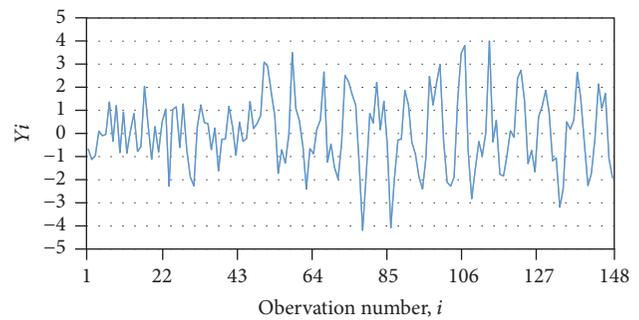


FIGURE 4: The SPC-EPC process outputs with the presence of a mixture type of SHI and CYC disturbances intruding after observation number 50, using the EPC adjustments.

Although many studies have investigated MCCPs identification, most of the existing works are concerned with the identification of the MCCPs for an SPC system alone. Only very few studies have focused on the CCP identification for a more sophisticated SPC-EPC process [6]. In [6], it was addressed that the five commonly observed disturbance patterns existed in an SPC-EPC process. The performances of the proposed extreme learning machine and random forest approaches were better than the approaches of ANN and rough set. In [6], the assumption of those five individual disturbances or their combined types of disturbances was consecutively occurred. That is, any two of the combined disturbances would not intrude into the SPC-EPC system at the same time, and they just occurred in a sequential manner. For a generic concern, this study considers the case where the combined disturbances can occur at the same time. Also, since seven types of combined disturbance need to be considered in this study, the typical single-stage classifiers hardly perform the classification tasks well. Accordingly, this study proposes the two-stage NN based classifiers to overcome the above-mentioned difficulties.

3.2. NN Classifier. NN can be referred to as one of the most widely used classifiers for practical applications [32]. Because the backpropagation neural network (BPNN) is widely used in many applications, this study employs a BPNN when

designing the ANN model. NN modeling can be briefly described as follows. The relationship between output (y) and inputs (x_1, x_2, \dots, x_a) in an ANN model can be formed as

$$y = \alpha_0 + \sum_{j=1}^b \alpha_j g \left(\delta_{0j} + \sum_{i=1}^a \delta_{ij} x_i \right) + \varepsilon, \quad (6)$$

where α_j ($j = 0, 1, 2, \dots, b$) and δ_{ij} ($i = 0, 1, 2, \dots, a$; $j = 0, 1, 2, \dots, b$) are the model connection weights; a is the number of input nodes; b is the number of hidden nodes; and ε is the error term.

In addition, a nonlinear functional mapping from the inputs (x_1, x_2, \dots, x_a) to the output y is performed by

$$y = f(x_1, x_2, \dots, x_a, w) + \varepsilon, \quad (7)$$

where w is a vector of the model parameters and f is a function determined by the NN structure and connection weights.

For NN structure, this study employs a logistic function to serve as the transfer function in the hidden layer, and the logical function is represented as

$$g(z) = \frac{1}{1 + \exp(-z)}. \quad (8)$$

3.3. SVM Classifier. In addition to using NN classifiers, SVM played an important role for CCPs recognition for a process. SVM modeling can be described as follows. For SVM modeling, there are two separable classes and sample data can be described as

$$D = \{(x_1, y_1), \dots, (x_l, y_l)\} \\ x_i \in \mathfrak{R}^n, y_i \in \{-1, 1\}, i = 1, 2, \dots, l \\ y_i = \begin{cases} 1, & \text{if } x_i \text{ in class 1} \\ -1, & \text{if } x_i \text{ in class 2,} \end{cases} \quad (9)$$

where l is the number of observations and n is the dimension of each observation. The decision function is given by $f(x) = \text{sign}(w^T x + b)$. The separating hyperplane is expressed as

$$f(x) = w^T x + b = 0, \quad (10)$$

where w is the coefficient vector and b is the constant.

To obtain the optimal hyperplane, we define the optimization problem as [33]

$$\min \Phi(\bar{x}) = \frac{1}{2} \|\bar{w}\|^2 \\ \text{s.t. } y_i (\bar{w}^T \bar{x}_i + b) \geq 1, \quad i = 1, 2, \dots, n. \quad (11)$$

It is difficult to solve (11), and we need to transform the optimization problem to the dual problem by Lagrange method. The value of α in the Lagrange method must be

nonnegative real coefficients. Equation (11) is transformed into the following constrained form [33]:

$$\max \Phi(\bar{w}, b, \xi, \alpha, \beta) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1, j=1}^N \alpha_i \alpha_j y_i y_j \bar{x}_i^T \bar{x}_j \\ \text{s.t. } \sum_{j=1}^n \alpha_j y_j = 0, \quad 0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, n. \quad (12)$$

In (12), C is the penalty factor and determines the degree of penalty assigned to an error.

In general, it could not find the linear separate hyperplane for all application data. For problems that cannot be linearly separated in the input space, the SVM uses the kernel method to transform the original input space into a high-dimensional feature space, where an optimal linear separating hyperplane can be found. The common kernel function are linear, polynomial, and radial basis function (RBF) and sigmoid. Since the RBF is one of the most widely used kernel functions, a RBF was used in this study. RBF is defined as [34]

$$K(\bar{x}_i, \bar{x}_j) = \exp(-\gamma \|\bar{x}_i - \bar{x}_j\|^2), \quad \gamma \geq 0, \quad (13)$$

where γ denotes the width of the RBF.

3.4. MARS Classifier. The MARS modeling is based on a divide-and-conquer strategy, where training datasets are partitioned into separate regions, each of which is assigned its own regression equation. The general MARS model can be described as follows [24]:

$$f(x) = b_0 + \sum_{m=1}^M b_m \prod_{j=1}^{J_m} [S_{j_m}(x_{\nu(j,m)} - l_{j_m})], \quad (14)$$

where b_0 and b_m are the parameters, M is the number of basis functions (BFs), J_m is the number of knots, S_{j_m} takes on values of either 1 or -1 and indicates the right or left sense of the associated step function, $\nu(j, m)$ is the label of the independent variable, and l_{j_m} is the knot location. The optimal MARS model is selected in a two-step procedure. The first step is to build a large number of BF to fit the data initially. The BF are deleted in the order of least contributions to the most, using the generalized cross-validation (GCV) criterion in the second step. The measure of variable importance is provided by observing the decrease in the calculated GCV values when a variable is removed from the model. The GCV is described as follows:

$$\text{GCV}(M) = \frac{(1/N) \sum_{i=1}^N [y_i - f_M(x_i)]^2}{[1 - C(M)/N]^2}, \quad (15)$$

where N is the number of observations and $C(M)$ are the cost penalty measures of a model containing M BF.

3.5. Research Framework. Figure 5 displays a generalized depiction of the research framework. As shown in Figure 5, we notice that considerable process disturbances could

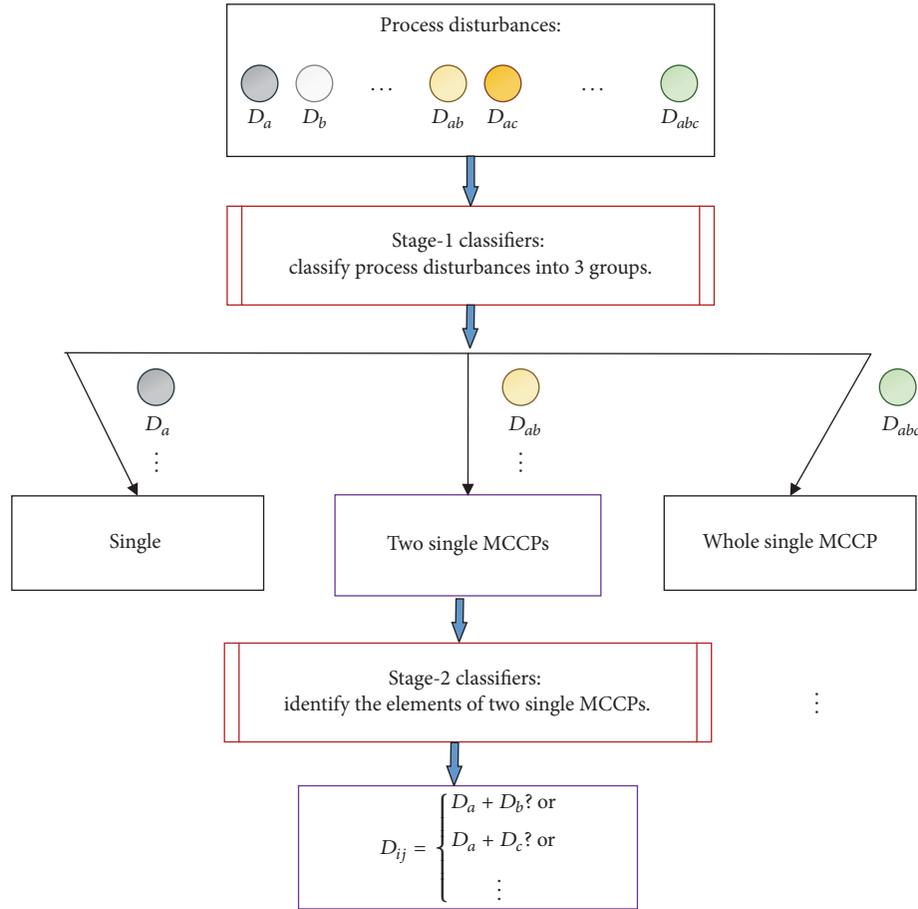


FIGURE 5: The research framework.

intrude into an SPC-EPC system. When disturbances intrude in the process, the process becomes unstable. In order to determine the root causes of the unstable process, we need to identify the patterns for the underlying disturbances. However, a classifier typically cannot perform the classification tasks well if the output variable has a large number of categories. Therefore, in stage-1, this study reduces the dimension of the output categories to three categories. These three categories of process disturbances include a single CCP, two single MCCPs, and whole single MCCP. Accordingly, the classifiers in stage-1 only need to classify fewer (i.e., three) categories of an output variable. Since the output variable of a classifier has fewer categories, the accuracy of classification can be greatly enhanced by using the classifiers in stage-1.

In addition, since this study focuses on identifying the two single MCCPs for the SPC-EPC process, the classifiers in stage-2 are used to identify elements (i.e., single disturbances) consisting of the two single MCCPs. Because fewer categories are associated with the output variables for stage-2 classifiers, the classifiers can have greater chance to maintain the high classification accuracy. Also, in this study, we employ three classifiers, ANN, SVM, and MARS, to perform the classification tasks in stage-1 and stage-2.

4. Experimental Results and Discussion

In this study, an industrial SPC-EPC process is assumed to be disturbed by five single disturbances that are described by (5). Also, since this study assumes that any 3 single disturbances may concurrently be intruded into the process, there are 7 kinds or categories that may need to be classified. For example, suppose that there are three single disturbances, CYC, SYS, and SHI, concurrently intruding into a process; Table 1 shows three types with seven categories of CCPs. Here, this study is mainly interested in identifying the two single mixture MCCPs for the SPC-EPC process. That is, this study focuses on identifying three MCCPs: {CYC, SYS}, {CYC, SHI}, and {SYS, SHI}.

In this study, the NN simulator Qnet97, which was developed by Vesta Services Inc., was used to develop the NN models. Qnet97 is a C-based simulator that provides a system for developing BPNN configurations by using a generalized delta learning algorithm. For SVM modeling, the package "e1071" in the R programming language was used in this study. The MARS model was constructed using MARS, which was developed by Salford Systems.

In addition, since seven categories are difficult to be correctly identified by using typical single-stage classifiers,

TABLE 1: Types of CCPs for 3 single disturbances in a process.

Types	Categories	Combination
Type I		
Single	C_1	{CYC}
Single	C_2	{SYS}
Single	C_3	{SHI}
Type II		
Two single MCCPs	C_4	{CYC, SYS}
Two single MCCPs	C_5	{CYC, SHI}
Two single MCCPs	C_6	{SYS, SHI}
Type III		
Whole single MCCP	C_7	{CYC, SYS, SHI}

this study proposes a two-stage identification mechanism to overcome the problem of considerable categories. The first stage of the proposed two-stage classifiers is used to initially identify three types of disturbances, type I, type II, and type III. Instead of identifying seven categories of disturbances, the classifiers in the first stage only need to identify three types of disturbances with the use of dimension reduction approach. Then, the second stage of the proposed classifiers is served to identify three categories of disturbances in type II. Since this study aims to obtain the AIR values for the three categories, C_4 , C_5 , and C_6 , we need to know the calculation of these AIR values. In this study, AIR is used for the classification performance measurement. AIR is defined as follows:

$$\text{AIR} = \frac{n_a}{N}, \quad (16)$$

where N is the total number of data vectors used in the identification process and n_a is the number of data vectors in N , where the true CCP type is accurately identified.

Considering the case of C_4 , we can have the AIR by using the following procedure. By using a certain classifier of the proposed mechanism, we can obtain the AIR values, denoted as A_1 and A_2 , in the first and second stages, respectively. The AIR for the case of C_4 is simply obtained by the multiplication of A_1 and A_2 . The AIR values for C_5 and C_6 can be easily obtained by using the same procedures.

In order to demonstrate the identification capability of the proposed approaches, this study performs a series of computer simulations. Suppose that an SPC-EPC process is represented by (4) and the parameters are arbitrarily selected as $q = 0.5$ and $\theta = 0.8$. In this study, we consider that three out of five single disturbances will intrude into an SPC-EPC system. Accordingly, we have seven combinations of CCPs for each set of concurrent three single disturbances. For example, by observing Table 1, we can have seven combinations for the concurrent three single disturbances. Based on the SPC-EPC system and disturbance models, this study generates the data vectors for the training and testing phases of the NN, SVM, and MARS classifiers. For the structures of these three classifiers, this study employs X (i.e., (3)) and Y (i.e., (4)) as the classifiers' inputs and considers Z (i.e., the classification category) as the classifiers' output. Since the typical classifiers

TABLE 2: The confusion matrix for the ANN classifier.

Predicted class	Actual class						
	0	1	2	3	4	5	6
0	20	6	0	0	0	0	10
1	200	47	0	0	0	0	0
2	36	223	55	70	43	0	55
3	5	24	245	186	93	0	100
4	6	0	0	44	164	0	120
5	33	0	0	0	0	70	15
6	0	0	0	0	0	230	0

would identify seven combinations, the values of Z are set to be from 0 to 6. The meanings of these values are described as follows:

$Z = 0$ represents the presence of the first combination (i.e., single {CYC}).

$Z = 1$ represents the presence of the second combination (i.e., single {SYS}).

$Z = 2$ represents the presence of the third combination (i.e., single {SHI}).

$Z = 3$ represents the presence of the fourth combination (i.e., MCCP {CYC-SYS}).

$Z = 4$ represents the presence of the fifth combination (i.e., MCCP {CYC-SHI}).

$Z = 5$ represents the presence of the sixth combination (i.e., MCCP {SYS-SHI}).

$Z = 6$ represents the presence of the seventh combination (i.e., whole single MCCP {CYC-SYS-SHI}).

Additionally, this study uses 4,900 and 2,100 data vectors for the training and testing phases, respectively. In the training phase, a set of 700 data vectors are generated from each combination. Consider Table 1 as an example. The first 700 data vectors are generated from the presence of {CYC} alone. The data vectors from 701 through 1,400 are generated from the presence of {SYS} alone. The same grouping is used up to the final data vectors from 4,201 through 4,900, which are generated from the combined presence of {CYC, SYS, SHI} disturbances. The testing data structure is similar to the training data structure. Specifically, the first 300 data vectors involve {CYC} disturbances alone and the final data vectors from 1,801 through 2,100 involve {CYC, SYS, SHI} disturbances. After performing the classification tasks with ANN, SVM, and MARS classifiers, we can compute the corresponding type I error and type II error rates. Tables 2, 3, and 4 present the corresponding confusion matrices for ANN, SVM, and MARS, respectively. For the identification results for all the categories in Table 1, type I error rates are 0.7419, 0.5767, and 0.4933 for ANN, SVM, and MARS, respectively. Also, type II error rates are 0.1237, 0.0961, and 0.0822 for ANN, SVM, and MARS, respectively. We can notice that all type I and type II error rates are not satisfactory.

In comparison to type I and type II errors, the AIR measure is easier to be understood by process personnel.

TABLE 3: The confusion matrix for the SVM classifier.

Predicted class	Actual class						
	0	1	2	3	4	5	6
0	151	201	1	64	44	1	77
1	137	99	0	5	0	0	0
2	1	0	260	15	33	0	36
3	2	0	0	88	17	5	2
4	5	0	25	54	156	8	0
5	3	0	0	43	12	272	0
6	1	0	14	31	38	14	185

TABLE 4: The confusion matrix for the MARS classifier.

Predicted class	Actual class						
	0	1	2	3	4	5	6
0	90	150	0	41	38	0	0
1	193	149	0	5	0	0	0
2	5	0	293	29	62	0	60
3	1	1	0	70	1	0	105
4	7	0	7	61	164	2	52
5	4	0	0	75	30	298	83
6	0	0	0	19	5	0	0

Additionally, since AIR was employed in [6], this study uses the AIR as a measure of accuracy for the various two-stage NN based classifiers presented in our study. Table 5 presents the identification results for all the CCPs in Table 1. The overall AIR values are 25.81%, 57.67%, and 50.76% for the ANN, SVM, and MARS classifiers, respectively. The AIR values of the three MCCPs (i.e., {CYC, SYS}, {CYC, SHI}, and {SYS, SHI}) are 46.47%, 57.33%, and 59.11% for the ANN, SVM, and MARS classifiers, respectively. These AIR values can be computed by using a confusion matrix. The overall AIR for the ANN model is computed by summing the diagonal elements and dividing the testing data vectors (i.e., 2,100) as follows:

$$\text{overall AIR} = \frac{\sum (20 + 47 + \dots + 0)}{300 \times 7} = 25.81\%. \quad (17)$$

Additionally, because the class values of 3, 4, and 5 in Table 2 represent the status of three MCCPs (i.e., {CYC, SYS}, {CYC, SHI}, and {SYS, SHI}), the AIR of the ANN model for the three MCCPs is computed as follows:

$$\text{AIR} = \frac{\sum (186 + 164 + 70)}{300 \times 3} = 46.67\%. \quad (18)$$

By using the same calculations, we can obtain the corresponding AIR values for the SVM and MARS models.

After performing ANN modeling, we found that a {2-6-1} topology with a learning rate of 0.01 provided the best results with the minimum testing RMSE. The notation $\{n_i-n_h-n_o\}$ represents the number of neurons in the input layer, hidden layer, and output layer, respectively. Because the RBF kernel

function is adopted in this study, the performance of the SVM is mainly affected by the values of two parameters (C and γ). There are no general rules for the choice of C and γ . In this study, the grid search proposed in [35] is used for parameter settings. The grid search method uses exponentially growing sequences of C and γ to identify good parameters (e.g., $C = 2^{-5}, 2^{-3}, 2^{-1}, \dots, 2^{15}$). The parameter settings for C and γ that generate the highest correct classification rate are considered to be ideal set. The trained SVM model with the best parameter settings, denoted as $\{C, \gamma\}$, is preserved and used during the monitoring stage for CCP recognition. Additionally, because there are no specific parameter settings for MARS, we simply denote the parameter settings as {null} for the MARS classifiers.

In a traditional design, the identification performance of the three classifiers is poor due to the fact that the classifier output Z contains too many categories. Therefore, we propose a two-stage mechanism in order to overcome the problems associated with a large number of categories.

Since identification performance of the typical design is not satisfactory when the output categories are considerably large, this study reduces the dimension of the output categories by using a two-stage mechanism. By considering Table 1, the first stage of the classifiers is used to identify three, instead of seven, types of disturbances (i.e., type I, type II, and type III). In the first stage, this study initially sets the values of classification categories Z as 0, 1, and 2, respectively. The value of 0 represents the presence of type I disturbances (i.e., single disturbance), the value of 1 represents the presence of type II disturbances (i.e., the MCCPs which we want to classify), and the value of 2 represents the presence of type III disturbances (i.e., the whole single CCP). This study also uses 4900 and 2100 data vectors for the training and testing phases. The second stage of the proposed design is to identify which set of MCCPs are presented in the underlying process. Namely, which one of {CYC, SYS}, {CYC, SHI}, and {SYS, SHI} existed in the system? In the second stage, since the process contains three combinations of MCCPs in type II, this study sets Z as three output values. "0" indicates that {CYC, SYS} is presented, "1" indicates that {CYC, SHI} is presented, and "2" indicates that the {SYS, SHI} mixture disturbance is presented in the process.

After performing the two-stage classification tasks, this study obtains the results which are listed in Table 6. In Table 6, the first column lists ten combinations for three out of five single disturbances. The second column presents the best two-stage classifiers associated with the parameter settings for the first and second stages, respectively. The last column in Table 6 shows the average AIR values. Observing Table 6, we can notice that the smallest average AIR value is 57.23%, occurring in the case of {SHI, TRE, CYC}. The possible reason may be the fact that the characteristics for the SHI, TRE, and CYC are similar, and those three disturbances cannot be effectively identified. In addition, all other average AIR values for the remaining nine combinations are greater than 60%. In general, the proposed BPNN-SVM and BPNN-BPNN classifiers possess satisfactory capability for identifying the MCCPs for an SPC-EPC process.

TABLE 5: Identification results for a single-stage design when {CYC, SYS, SHI} exist concurrently in an SPC-EPC process.

Classifier {parameter settings}	Overall AIR	AIR for three MCCPs
ANN {2, 6, 1}	25.81%	46.67%
SVM {2 ⁻² , 2 ⁵ }	57.67%	57.33%
MARS {null}	50.67%	59.11%

TABLE 6: Identification results for all combinations of MCCPs in a two-stage design.

Combinations	Best two-stage classifiers [{{parameters}}, {{parameters}}]	Average AIR
{CYC, SYS, STR}	BPNN-SVM [{{2, 2, 1}}, {{0.0625, 0.5}}]	73.75%
{SHI, SYS, CYC}	SVM-SVM [{{0.0625, 8}}, {{0.5, 8}}]	62.89%
{SHI, SYS, STR}	BPNN -SVM [{{2, 5, 1}}, {{2, 4}}]	84.89%
{SHI, CYC, STR}	BPNN -MARS [{{2, 6, 1}}, {{null}}]	80.69%
{TRE, SYS, CYC}	BPNN -SVM [{{2, 5, 1}}, {{0.5, 32}}]	60.58%
{TRE, SYS, STR}	BPNN -MARS [{{2, 5, 1}}, {{null}}]	73.89%
{TRE, CYC, STR}	BPNN - BPNN [{{2, 3, 1}}, {{2, 6, 1}}]	77.72%
{SHI, TRE, CYC}	BPNN -SVM [{{2, 4, 1}}, {{1, 8}}]	57.23%
{SHI, TRE, SYS}	BPNN - BPNN [{{2, 5, 1}}, {{2, 6, 1}}]	79.22%
{SHI, TRE, STR}	BPNN - BPNN [{{2, 3, 1}}, {{2, 6, 1}}]	61.09%

5. Conclusion

CCP identification is crucial for the improvement of industrial processes. Because an integrated mechanism using SPC and EPC can result in very effective monitoring and controlling performance, we must focus heavily on CCP identification for such a process. Thus, the purpose of this study was to identify MCCPs for an SPC-EPC system. Additionally, we proposed a two-stage classification technique in order to overcome the problems associated with a large number of output categories. The first stage of the proposed approach employs classifiers to identify a reduced number of output categories and the second-stage classifiers are used to effectively determine the types of MCCPs for a process.

The performance of the proposed two-stage classification technique was verified through a series of computer experiments. The proposed BPNN-SVM and BPNN-BPNN models achieve satisfactory performance for identifying the MCCPs for an SPC-EPC process. In our study, we used the AIR to measure accuracy for various two-stage NN based classifiers. Another measurement, area under the receiver operating characteristic (ROC) curve (AUC), could also be used to measure the accuracy of various classifiers. One limitation of such a measure is that AUC values are difficult to calculate for our proposed two-stage models. One possible future research

direction is to compute AUC values for our two-stage models. Another limitation for the proposed two-stage classifiers is the computational time. A two-stage classifier may require more time to obtain the classification results. Faster computer systems are suggested to perform the two-stage classification tasks, since they would help to speed up the process.

Additionally, this study considered two single MCCPs identification. An attempt to classify three single or even four single MCCPs would be a valuable contribution to future research. Some other classifiers, such as multivariate adaptive regression splines (MARS) and random forests may also be employed to identify the mixture disturbance patterns for a multivariate SPC-EPC system.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work is partially supported by the Ministry of Science and Technology of the Republic of China (Grant no. MOST 106-2221-E-030-010-MY2).

References

- [1] J. F. MacGregor, T. J. Harris, and J. D. Wright, "Duality between the control of processes subject to randomly occurring deterministic disturbances and arima stochastic disturbances," *Technometrics*, vol. 26, no. 4, pp. 389–397, 1984.
- [2] G. Box and T. Kramer, "Statistical process monitoring and feedback adjustment—a discussion," *Technometrics*, vol. 34, no. 3, pp. 251–285, 1992.
- [3] D. C. Montgomery and C. M. Mastrangelo, "Some statistical process control for autocorrelation data (with discussion)," *Journal of Quality Technology*, vol. 23, no. 3, pp. 179–193, 1991.
- [4] Y. E. Shao, "Integrated application of the cumulative score control chart and engineering process control," *Statistica Sinica*, vol. 8, no. 1, pp. 239–252, 1998.
- [5] W. A. Shewhart, *Economic Control of Quality of Manufactured Product*, D. Van Nostrand Company, Inc., New York, NY, USA, 1931.
- [6] Y. E. Shao and C.-C. Chiu, "Applying emerging soft computing approaches to control chart pattern recognition for an SPC-EPC process," *Neurocomputing*, vol. 201, pp. 19–28, 2016.
- [7] C.-J. Lu, Y. E. Shao, and P.-H. Li, "Mixture control chart patterns recognition using independent component analysis and support vector machine," *Neurocomputing*, vol. 74, no. 11, pp. 1908–1914, 2011.
- [8] R.-S. Guh and J. D. T. Tannock, "Recognition of control chart concurrent patterns using a neural network approach," *International Journal of Production Research*, vol. 37, no. 8, pp. 1743–1765, 1999.
- [9] S. A. Lesany, A. Koochakzadeh, and S. M. T. Fatemi Ghomi, "Recognition and classification of single and concurrent unnatural patterns in control charts via neural networks and fitted line of samples," *International Journal of Production Research*, vol. 52, no. 6, pp. 1771–1786, 2014.
- [10] W.-A. Yang and W. Zhou, "Autoregressive coefficient-invariant control chart pattern recognition in autocorrelated manufacturing processes using neural network ensemble," *Journal of Intelligent Manufacturing*, vol. 26, no. 6, pp. 1161–1180, 2015.
- [11] R.-S. Guh, "Real-time recognition of control chart patterns in autocorrelated processes using a learning vector quantization network-based approach," *International Journal of Production Research*, vol. 46, no. 14, pp. 3959–3991, 2008.
- [12] V. Ranaee and A. Ebrahimzadeh, "Control chart pattern recognition using neural networks and efficient features: a comparative study," *Pattern Analysis and Applications*, vol. 16, no. 3, pp. 321–332, 2013.
- [13] A. Ebrahimzadeh, J. Addeh, and V. Ranaee, "Recognition of control chart patterns using an intelligent technique," *Applied Soft Computing*, vol. 13, no. 5, pp. 2970–2980, 2013.
- [14] L.-J. Kao, T.-S. Lee, and C.-J. Lu, "A multi-stage control chart pattern recognition scheme based on independent component analysis and support vector machine," *Journal of Intelligent Manufacturing*, vol. 27, no. 3, pp. 653–664, 2016.
- [15] H. De La Torre Gutierrez and D. T. Pham, "Estimation and generation of training patterns for control chart pattern recognition," *Computers & Industrial Engineering*, vol. 95, pp. 72–82, 2016.
- [16] P. Xanthopoulos and T. Razzaghi, "A weighted support vector machine method for control chart pattern recognition," *Computers & Industrial Engineering*, vol. 70, no. 1, pp. 134–149, 2014.
- [17] C.-H. Wang and W. Kuo, "Identification of control chart patterns using wavelet filtering and robust fuzzy clustering," *Journal of Intelligent Manufacturing*, vol. 18, no. 3, pp. 343–350, 2007.
- [18] W.-A. Yang, W. Zhou, W. Liao, and Y. Guo, "Identification and quantification of concurrent control chart patterns using extreme-point symmetric mode decomposition and extreme learning machines," *Neurocomputing*, vol. 147, no. 1, pp. 260–270, 2015.
- [19] M. Salehi, A. Bahreininejad, and I. Nakhai, "On-line analysis of out-of-control signals in multivariate manufacturing processes using a hybrid learning-based model," *Neurocomputing*, vol. 74, no. 12–13, pp. 2083–2095, 2011.
- [20] S. K. Gauri and S. Chakraborty, "Recognition of control chart patterns using improved selection of features," *Computer & Industrial Engineering*, vol. 56, pp. 1577–1588, 2009.
- [21] S. C. Du, D. L. Huang, and J. Lv, "Recognition of concurrent control chart patterns using wavelet transform decomposition and multiclass support vector machines," *Computers & Industrial Engineering*, vol. 66, no. 4, pp. 683–695, 2013.
- [22] N. Gu, Z. Cao, L. Xie, D. Creighton, M. Tan, and S. Nahavandi, "Identification of concurrent control chart patterns with singular spectrum analysis and learning vector quantization," *Journal of Intelligent Manufacturing*, vol. 24, no. 6, pp. 1241–1252, 2013.
- [23] W. Hachicha and A. Ghorbel, "A survey of control-chart pattern-recognition literature (1991–2010) based on a new conceptual classification scheme," *Computers & Industrial Engineering*, vol. 63, no. 1, pp. 204–222, 2012.
- [24] J. H. Friedman, "Multivariate adaptive regression splines," *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991.
- [25] S. Psarakis, "The use of neural networks in statistical process control charts," *Quality and Reliability Engineering International*, vol. 27, no. 5, pp. 641–650, 2011.
- [26] F. Kadri, F. Harrou, S. Chaabane, Y. Sun, and C. Tahon, "Seasonal ARMA-based SPC charts for anomaly detection: application to emergency department systems," *Neurocomputing*, vol. 173, pp. 2102–2114, 2016.
- [27] M. Weese, W. Martinez, F. M. Megahed, and L. A. Jones-Farmer, "Statistical learning methods applied to process monitoring: an overview and perspective," *Journal of Quality Technology*, vol. 48, no. 1, pp. 4–27, 2016.
- [28] S. K. Gauri and S. Chakraborty, "Feature-based recognition of control chart patterns," *Computers & Industrial Engineering*, vol. 51, no. 4, pp. 726–742, 2006.
- [29] S. Y. Lin, R. S. Guh, and Y. R. Shiue, "Effective recognition of control chart patterns in autocorrelated data using a support vector machine based approach," *Computers & Industrial Engineering*, vol. 61, pp. 1123–1134, 2011.
- [30] S. W. Roberts, "Control chart tests based on geometric moving averages," *Technometrics*, vol. 1, no. 3, pp. 239–250, 1959.
- [31] E. S. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, no. 1–2, pp. 100–115, 1954.
- [32] Y. E. Shao, C.-D. Hou, and C.-C. Chiu, "Hybrid intelligent modeling schemes for heart disease classification," *Applied Soft Computing*, vol. 14, pp. 47–52, 2014.
- [33] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, Berlin, Germany, 2000.
- [34] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural Networks*, vol. 17, no. 1, pp. 113–126, 2004.
- [35] C. W. Hsu, C. C. Chang, and C. J. Lin, "A practical guide to support vector classification," Tech. Rep., Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan, 2003.

Research Article

A Gain-Scheduling PI Control Based on Neural Networks

Stefania Tronci and Roberto Baratti

*Dipartimento di Ingegneria Meccanica, Chimica e dei Materiali, Università degli Studi di Cagliari,
Via Marengo 2, 09123 Cagliari, Italy*

Correspondence should be addressed to Stefania Tronci; stefania.tronci@dimcm.unica.it

Received 13 July 2017; Revised 19 September 2017; Accepted 24 September 2017; Published 19 October 2017

Academic Editor: Jing Na

Copyright © 2017 Stefania Tronci and Roberto Baratti. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents a gain-scheduling design technique that relies upon neural models to approximate plant behaviour. The controller design is based on generic model control (GMC) formalisms and linearization of the neural model of the process. As a result, a PI controller action is obtained, where the gain depends on the state of the system and is adapted instantaneously on-line. The algorithm is tested on a nonisothermal continuous stirred tank reactor (CSTR), considering both single-input single-output (SISO) and multi-input multi-output (MIMO) control problems. Simulation results show that the proposed controller provides satisfactory performance during set-point changes and disturbance rejection.

1. Introduction

Most industrial plants are nonlinear in nature, but process control often relies on traditional linear PID algorithm, because of its simplicity and well recognition by the industry. This solution can be justified by the fact that nonlinear processes can be approximated with a linear model as they approach steady state; therefore, in case of close regulatory control, the use of a linear control is adequate [1]. Nevertheless, for those nonlinear processes whose nonlinearities are strong and large changes of the operating conditions occur during the operation, linear controller designs are inadequate and more effective alternatives should be considered [2]. The availability of powerful computer tools opened the way to implement advanced process control strategies, where system nonlinearities can be taken into account.

Model-based feedback control can be a valid alternative to the use of the linear model-free PI algorithm, and the use of adaptive parameters along the process motions can represent a possible solution for controlling nonlinear time-variant processes [3]. Adaptive systems are also used to compensate input delay, as proposed in Na et al. [4] where an adaptive NN observer was designed for nonlinear systems. The generic model control (GMC) of Lee and Sullivan [5] is probably one of the simplest nonlinear control techniques to install and

maintain among nonlinear model-based controllers. In the chemical engineering field the application of GMC strategy was investigated for control reactors [6, 7], batch cooling crystallizers [8], batch and semibatch polymerization reactors [9, 10], and, recently, multistage flash desalinations [11].

In this work, the GMC algorithm was used in conjunction with a dynamic neural network model, which describes the nonlinear relationship between controlled outputs and manipulated inputs [12, 13]. The main advantage of the proposed algorithm is the obtainment of a PID-like controller structure, where the nonlinear dependence of the process gain on the operating condition is achieved by a gain-scheduling control scheme. The proposed algorithm is simple to implement and it can be obtained without the need of a detailed knowledge of the plant; therefore it is suited for industrial applications where standard solutions are generally preferred. The performance of the proposed technique for nonlinear process control was tested for two case studies, referring to a SISO [14] and a MIMO [15] control problem for a nonisothermal continuously stirred tank reactor (CSTR). These two cases, which are well known benchmarks for testing control methodology, were selected because the strong nonlinearities, due to the Arrhenius dependence of the kinetic rate on temperature, lead to a very rapid response of the process variables in regions of high conversion and a very

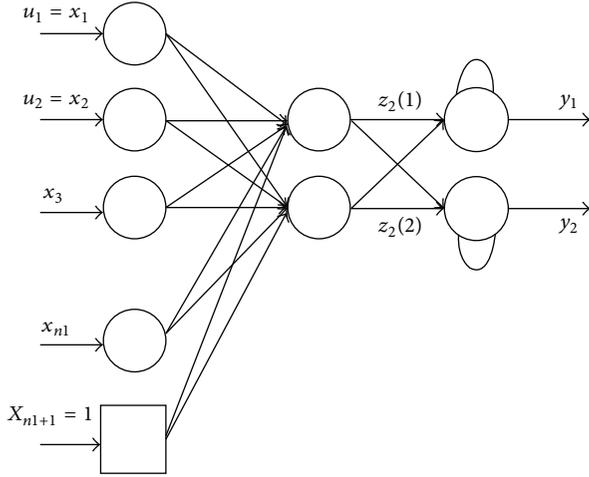


FIGURE 1: Neural network structure.

mild response in regions of low conversion. Therefore, the need for adaptive control strategies is more evident.

2. Neural Network Model

There are several possibilities of building a dynamic neural network that may be classified in two main classes: time-lagged feedforward networks, TLFN, or recurrent neural networks, RNN (cf. [16]). While in the TLFN the dynamics are generally accounted for in the input layer as a linear combination of inputs (present and past values of the inputs) followed by a static nonlinear mapper between inputs and outputs, in the RNN the memory mechanism is brought inside the nonlinear mapping; that is, recurrent connections are applied among some or all layers (cf. Jordan or Elman networks). Both the architectures are able to describe the dynamic behaviour of a process, while training procedure and stability become an issue moving from the TLFN to the RNN (cf. [16]).

In this work, a multi-feed-forward neural network with recurrent neurons in the output layer is used to describe the process dynamics. The net topology is shown in Figure 1 and can be considered a special case of the TLFN where the dynamic is moved from the input to the output layer and, in this way, the advantage of the TLFN over the RNN is maintained.

The equations that describe the neuron output evolution are reported in the following:

$$\tau \frac{dy_k}{dt} + y_k = \sum_{j=1}^{n_2} w_2(j, k) z_2(j), \quad (1a)$$

$$z_2(j) = f \left(\sum_{i=1}^{n_1+1} w_1(i, j) z_1(i) \right), \quad (1b)$$

where the activation function $f(\cdot)$ is chosen as

$$f(x) = \frac{1}{1 + \exp(-x)}, \quad x \in \mathcal{R}. \quad (2)$$

The weight $w_2(j, k)$ is the interconnection between the k th output and the j th hidden neuron; $w_1(i, j)$ is the connection between the i th input and the j th hidden neuron; \mathbf{z}_2 is the output vector from the hidden layer; and \mathbf{z}_1 is the input vector. It is worth noting that the nonlinear plant characteristic modelled by (1a)-(1b) is stored in the weights between the input and output layers, and this represents the long-term capability prediction, while the locally recurrent neurons in the output layer may be thought of as a representational layer for time (short-term) of information (cf. [16]).

In the present work, this neural model will be applied to develop a gain-scheduling control scheme. In order to obtain the simplest controller structure, a number of hidden neurons equal to the number of manipulated variables are selected. As results reported in the following sections will demonstrate, this choice does not affect the neural model performance. Under this assumption, for a SISO system the neural network expression becomes

$$\tau \frac{dy}{dt} + y = w_2(1, 1) z_2(1) = w_2(1, 1) f(u, \mathbf{x}), \quad (3)$$

where y is the predicted output and u is the manipulated variable. The variable $z_2(1)$ is the only output of the hidden neuron, which nonlinearly depends on the unique manipulated variable, u , since sigmoidal activation function is used for neurons in the hidden layer. The vector \mathbf{x} represents the other net inputs necessary to describe the plant properly. For the sake of brevity, the case of the neural network for a MIMO system is not reported, since it can be straightforwardly derived from (3).

3. Neural Based Gain-Scheduling Control

Starting from the nonlinear neural model, a GMC [5] approach is used to synthesize the gain-scheduling control. In order to define the controller action, the desired output behaviour has to be specified in the form of a trajectory. Then the process model is used directly to synthesize the controller required to cause the process output to follow this trajectory. A good choice [5] for the reference trajectory y_r is reported in

$$\frac{dy_r}{dt} = K_1 (y_d - y) + K_2 \int (y_d - y) dt, \quad (4)$$

where, for a given desired output y_d , a suitable selection of parameters K_1 and K_2 can be made to achieve a variety of responses in $y(t)$. Substituting the reference trajectory in (3), the action of the manipulated variable can be implicitly derived from the model. Of course, the control action in this case is not linear, and it is difficult to implement. Referring to Ogunnaike and Ray [17], a GMC control structure can lead to a PI-like controller if a first-order system is driven to follow a first-order reference trajectory, that is, setting $K_2 = 0$ in (4).

Letting $y = y_r$, the process model (3) can be used directly to obtain the controller required to cause the process to follow trajectory (4) and solving for $u(t)$. It should be pointed out that using model (3) no explicit solution is available for the manipulated input and the controller equation has

to be solved numerically. The controller algorithm can be simplified by linearizing the right-hand side of (3); hence the model for a SISO system becomes

$$\tau \frac{dy}{dt} + y = w_2(1, 1) \frac{\partial z_2}{\partial u} u(t). \quad (5)$$

In the following, the coefficient of the manipulated input will be indicated with K (6). The gain K depends on the state of the system, through the derivative of z_2 with respect to the input of the neural model:

$$K = w_2(1, 1) \frac{\partial z_2}{\partial u}. \quad (6)$$

The nonlinearity of the system is taken into account by calculating at every sampling time the derivative of z_2 with respect to the inputs from the neural model (3).

The PI controller action is derived at this point by setting $K_2 = 0$ in the reference trajectory (4), leading to

$$\frac{dy_r}{dt} = K_1 (y_d - y), \quad (7)$$

or

$$\frac{dy_r}{dt} = K_1 e(t), \quad (8)$$

where $e(t)$ represents the usual feedback error term. Integrating (8), the following expression for the desired trajectory is obtained:

$$y_r = K_1 \int e(t) dt. \quad (9)$$

The control action required to cause the process to follow the trajectory described by (8) and (9) may be obtained by substituting the desired trajectory expressions in the linearized model (5), leading to

$$\tau K_1 e(t) + K_1 \int e(t) dt = K u(t) \quad (10)$$

and then solving for $u(t)$:

$$u(t) = \frac{\tau K_1}{K} \left[e(t) + \frac{1}{\tau} \int_0^t e(\theta) d\theta \right]. \quad (11)$$

In this way, a PI controller law is obtained, where e represents the actual error, τ is equal to the time constant of the output neuron, K_1 is a tuning parameter and is set as the inverse of the desired time of response, and K is the static gain of the linear first-order system that we have obtained by linearizing the neural model of the system, as defined by (5). The gain is a function of the system status because it depends on the derivative of z_2 with respect to the input. The presence of the integral term will compensate for the error of the approximated neural model.

The advantage of such an approach is that the nonlinear behaviour of the process is considered because of the time-varying nature of the controller gain. As a result, since parameters are simply adjusted on-line for all process conditions, a

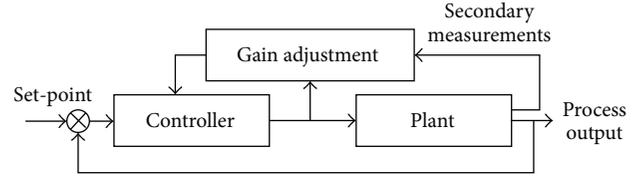


FIGURE 2: Neural scheduled adaptive control scheme.

standard PI controller structure is maintained. The proposed methodology has the scheme of a gain-scheduling control approach, as the block diagram in Figure 2 shows. It is important to note that in this case the use of a neural model removes the need for detailed process knowledge to define operating bands and for open loop tests to locally calibrate the controller gain within each band, which is the typical drawback of the gain-scheduling controller scheme.

4. Results and Discussion

As mentioned in Introduction, two continuously stirred tank reactor systems are considered to show the performance of the proposed control algorithm. The two proposed case studies are well known benchmarks for advanced control methodology testing [14, 15, 18].

4.1. Case 1. The first case study is the CSTR proposed by Lightbody and Irwin [18] and Ge et al. [14]. The system consists of a constant volume reactor cooled by a single coolant stream, and the objective is to control the effluent concentration, y_1 , by manipulating the coolant flow rate, u_1 . The model equations are reported as follows:

$$\frac{dC_a}{dt} = \frac{q}{V} (C_{a0} - C_a) - a_0 C_a e^{-E/RT_a}, \quad (12a)$$

$$\begin{aligned} \frac{dT_a}{dt} = \frac{q}{V} (T_f - T_a) + a_1 C_a e^{-E/RT_a} \\ + a_3 q_c (1 - e^{-a_2/q_c}) (T_{cf} - T_a), \end{aligned} \quad (12b)$$

where $y_1 = C_a$ and $u_1 = q_c$. The model parameters and nominal values used for this work are reported in Table 1.

The selected neural network model that describes the process is composed of two neurons in the input layer, one neuron in the hidden one (because one is the number of the manipulated variables), and one neuron in the output layer. The system has one natural input, which is the coolant flow rate, u_1 . In order to provide more information on the state of the reactor, another input, x_2 , is selected as energy balance around the cooling system. The net is trained using data obtained simulating the reactor for 300 minutes (1500 points sampled every 0.2 minutes), by exciting the plant through step variations of the manipulated variable. The input was modified every 5 minutes, randomly varying the amplitude of the step in the range 94.7–108.0 l/min. The Levenberg-Marquardt algorithm is used to estimate the network's weights, with the sum of squared error as objective function and performing the training for one hundred initial

TABLE 1: Summary of parameters and variables in the CSTR model for Case 1.

Symbol	Nominal Value	Description
q	100 ml/min	Process flow-rate
C_{a0}	1 mol/l	Concentration of component A
T_f	350 K	Feed temperature
T_{cf}	350 K	Inlet coolant temperature
q_c	100 l/min	Coolant flow-rate
V	100 ml	Volume of tank
h_a	$7 \cdot 10^5$ J/(min K)	Heat transfer coefficient
a_0	$7.2 \cdot 10^{10}$ min ⁻¹	Pre exponential factor
E/R	$1 \cdot 10^{10}$ K	Activation energy
$-\Delta H$	$2 \cdot 10^4$ cal/mol	Heat of reaction
ρ_1, ρ_c	$1 \cdot 10^3$ g/l	Liquid densities
C_p, C_{pc}	1 cal/(g K)	Heat capacities
$a_1 = \frac{(-\Delta H) a_0}{\rho_1 C_p} = 1.44 \cdot 10^{13}$	$a_2 = \frac{h_a}{\rho_c C_{pc}} = 698.7$	$a_3 = \frac{\rho_c C_{pc}}{\rho_1 C_p V} = 0.01$

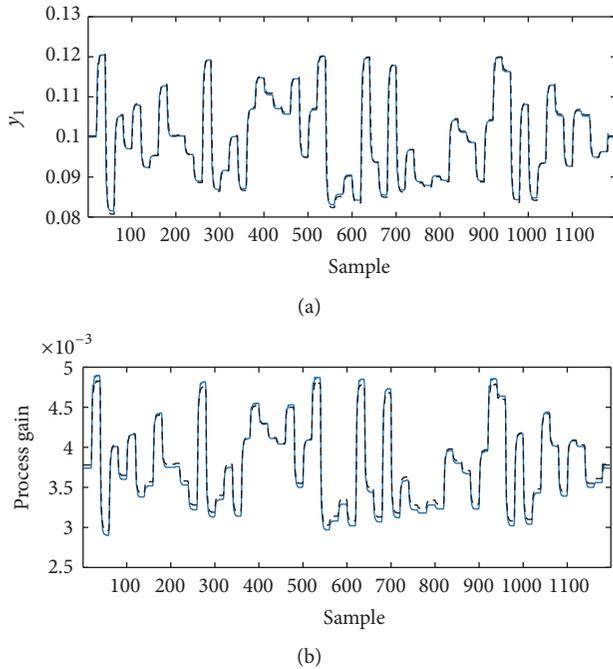


FIGURE 3: Performance of the neural model to reconstruct concentration and Jacobian for CSTR in Case 1. (a) Concentration predicted by the neural model (black dashed line) and CSTR (blue line). (b) Process gain calculated with the neural model (black dashed line) and actual gain of the process (blue line).

values, randomly generated. The selected weights are the ones leading to the lower error calculated on validation set, which is the 30% of the total amount of data recorded.

The capability of the network to reconstruct the system dynamics is shown in Figure 3, where concentration and Jacobian values estimated by the neural model are compared to those calculated by integrating equations ((12a)-(12b)) of the true plant. In this case, the output was obtained by randomly changing the manipulated variable and corrupting

the measured inputs to the network by noise, in particular introducing an error of $\pm 1^\circ\text{C}$ in temperature measurements and ± 2 l/min in the coolant flow rate measurement. Results indicate that the neural model reliably represents the system behaviour, at least for the considered process conditions; in fact the line representing the estimated variables completely masks the true values. When the plant model is not available, proof of the ability of the network to reconstruct system gain can be accomplished experimentally by performing an appropriate set of step tests.

The obtained dynamic neural model was then applied to construct the gain-scheduling PI controller described in Section 3. As a result, the only parameter to be tuned is the inverse of the desired time response, K_1 , because the controller gain and the integral time are derived from the model (the latter is not adjusted in time). In order to guarantee the robustness of the control system, the inverse of K_1 is set 2.5 times smaller than the maximum characteristic time of the process, applying the recommended choice for model-based controller that suggests to have the desired closed-loop time constant for a first-order process greater than 0.2τ [17]. The good fit shown in Figure 3 between the Jacobian calculated by the neural model and the true one guarantees that the gain will be properly adapted on-line.

The adaptive controller technique was tested in terms of set-point tracking, performing the test and comparing the results to a conventional PI controller, as reported in Ge et al. [14]. The set-point tracking results are shown in Figure 4, where the output variable, y_1 , and the loads on the manipulated variables, u_1 , are reported for the adaptive controller and the conventional PI. The results show that the adaptive controller exhibits good set-point tracking capabilities (short response time), without requiring excessive loads on the manipulated variables. Indeed, the curves representing the programmed set-point changes are almost completely masked by the ones representing the dynamic behaviour of the CSTR under the control of the gain scheduled PI. The presence of a low overshoot is the compromise for a short time response. On the other hand, the conventional PI

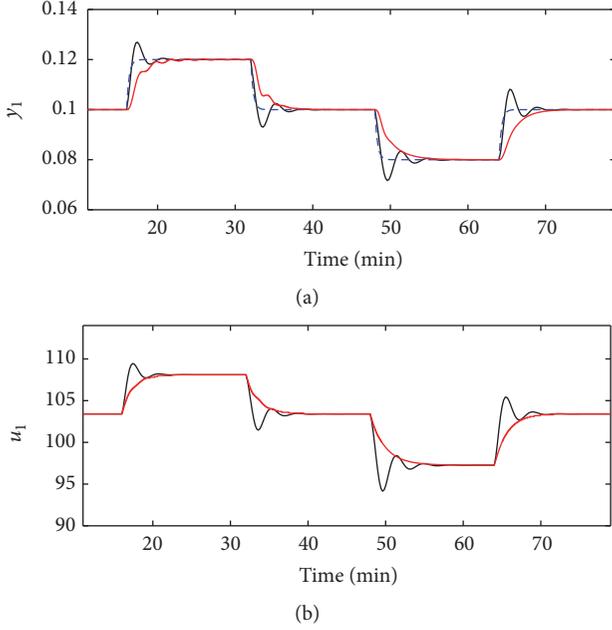


FIGURE 4: Performance of the gain-scheduling PI in Case 1 and a conventional PI with $K_c = 44.0$ and $\tau_I = 0.25$. (a) Set-point (dashed blue line), gain-scheduling controller (black line), and conventional PI (red line). (b) Manipulated variable for gain-scheduling controller (black line) and conventional PI (red line).

shows a sluggish response with respect to the gain-scheduling controller.

4.2. *Case 2.* As a second case study, a CSTR in which an exothermic first-order reaction takes place is considered. This benchmark was proposed by Scott and Ray [15] in order to demonstrate the inadequacy of a standard PI controller in such nonlinear systems and is described in a dimensionless form by the following differential equations:

$$\frac{dx_1}{dt} = \frac{1}{\tau} \left(-(m_2 + 1)(x_1 - d_2) + Da(1 - x_1)e^{yx_2/(x_2+1)} \right), \quad (13a)$$

$$\frac{dx_2}{dt} = \frac{1}{\tau} \left((m_2 + 1)(d_1 - x_2) + \frac{BDa}{\gamma} (1 - x_1)e^{yx_2/(x_2+1)} - \beta(x_2 - m_1) \right). \quad (13b)$$

According to Scott and Ray [15], the meaning of the constants and variables that appear in (13a) and (13b), along with the normalized variable used for training the network and representing the results of the control system, are summarized in Table 2.

The control objective is to maintain both bulk temperature and concentration at set-point values. The manipulated variables are the coolant temperature, m_1 , and the input feed flow rate, m_2 , while the control variables are the reactant mole fraction, x_1 , and the bulk temperature, x_2 .

The dynamic neural model of the CSTR consists of a net with four neurons in the input layer, two neurons in the hidden layer, and two neurons in the output layer. Four inputs were selected with the aim of giving the net the most representative information about the process status. Indeed, two terms which represent, respectively, the heat flux exchange through the cooling system and a measure of the heat flux carried out with the convective outlet stream were fed as input to the network, along with the two manipulated variables (cf. [13]).

The dynamic neural network was trained using 1000 data points generated through the CSTR dynamic simulator (sampling time equal to 0.1 min), randomly changing the dimensionless manipulated variable. The Levenberg-Marquardt algorithm was used to calculate the weights using the sum of the squared error as objective function and selecting the best model evaluating the predictions on the validation test. Also in this situation, it was verified that the Jacobian of the neural model fitted the exact one, which is a guarantee of the robustness of the control system. Results are not reported for the sake of brevity.

As in the previous case, the only controller parameter to be tuned is the inverse of the desired time response, K_1 , because the controller gain and the integral time are derived from the model. In this case, the inverse of K_1 is set 2.5 times smaller than the characteristic time of the process for both the controllers [17].

The gain-scheduling PI controller was tested in terms of set-point tracking and disturbance rejection capabilities, performing the same test as in Scott and Ray [15]. The set-point changes and disturbances imposed on the system to test the performance of the proposed methodology are summarized in Table 3.

For the sake of brevity, only the performance of the gain-scheduling control structure at the point of maximum gain is reported in graphical form, while all the other results are reported in Table 4 using the integral squared error (ISE) as performance index:

$$ISE = \int_0^{\infty} (y_{sp,1} - y_1)^2 dt + \int_0^{\infty} (y_{sp,2} - y_2)^2 dt. \quad (14)$$

The set-point tracking results are shown in Figure 5(b) for the first variable, y_1 , and in Figure 5(c) for the second variable, y_2 , while the loads on the manipulated variables are reported in Figure 5(a). The results show that the controller exhibits good set-point tracking capability (short response time) without requiring excessive loads on the manipulated variables. Indeed, the curves representing the programmed set-point changes (dashed curves) are completely masked by the ones representing the dynamic behaviour of the CSTR under the control of the gain scheduled PI controller. The performance of the controller with respect to disturbance rejection is shown in Figures 5(d)–5(f). Also in this situation the gain-scheduling controller performs well and exhibits good capability to quickly compensate for the upsets entering the CSTR.

As in the previous case, noisy measured inputs fed to the neural network did not affect the performance of the

TABLE 2: Summary of parameters and variables in the CSTR model for Case 2.

Symbol	Value or Range	Description
$\tau = \frac{V}{F_0}$	1.0	Nominal space time of reactor
$Da = \frac{k_0 e^{-\gamma} V}{F_0}$	0.11	Damkohler number
$\gamma = \frac{E}{(RT_{f0})}$	20	Dimensionless activation energy
$B = \frac{(-\Delta H)c_{A0}\gamma}{\rho C_p T_{f0}}$	7.0	Dimensionless heat of reaction
$\beta = \frac{hA}{\rho C_p F_0}$	0.5	Dimensionless transfer coefficient
F_0	1.0	Nominal feed rate
T_{f0}	300	Nominal feed temperature
c_{A0}	1.0	Nominal feed concentration
$x_1 = \frac{c_{A0} - c_A}{c_{A0}}$		Outflow concentration
$x_2 = \frac{T - T_{f0}}{T_{f0}}$		Outflow temperature
$m_1 = \frac{T_c - T_{f0}}{T_{f0}}$	$T_c \in [250, 350]$	Coolant temperature
$m_2 = \frac{F - F_0}{F_0}$	$T_c \in [0.5, 1.5]$	Input feed rate
$d_1 = \frac{T_f - T_{f0}}{T_{f0}}$	$T_f \in [295, 305]$	Feed temperature
$d_2 = \frac{c_{A0} - c_{Af}}{c_{A0}}$	$c_{Af} \in [0.9, 1.1]$	Feed concentration
Scaled inputs or outputs		
$y_1 = \frac{c_A - 0.755}{0.65}$	$y_1 \in [-1, 1]$	Outflow concentration
$y_2 = \frac{T - 317.1}{65}$	$y_2 \in [-1, 1]$	Outflow temperature
$u_1 = \frac{T_c - 300}{50}$	$u_1 \in [-1, 1]$	Coolant temperature
$u_2 = \frac{F - 1}{0.5}$	$u_2 \in [-1, 1]$	Input feed rate

TABLE 3: Programmed plant tests for Case 2.

Schedules	Set-point changes	Disturbance changes	Time
Nominal (0.000, 0.000)	$\Delta_1 \mathbf{y} = (0.000, -0.150)$ $\Delta_2 \mathbf{y} = (-0.150, 0.000)$	$\Delta_1 \mathbf{d} = (0.700, 0.000)$	@Time = (50, 100)
		$\Delta_2 \mathbf{d} = (-0.700, 0.000)$	@Time = (150, 200)
		$\Delta_3 \mathbf{d} = (0.000, 0.700)$	@Time = (250, 300)
		$\Delta_4 \mathbf{d} = (0.000, -0.700)$	@Time = (350, 400)
Maximum gain (-0.303, 0.242)	$\Delta_1 \mathbf{y} = (0.000, -0.150)$ $\Delta_2 \mathbf{y} = (-0.150, 0.000)$	$\Delta_1 \mathbf{d} = (0.700, 0.000)$	@Time = (50, 100)
		$\Delta_2 \mathbf{d} = (-0.700, 0.000)$	@Time = (150, 200)
		$\Delta_3 \mathbf{d} = (0.000, 0.700)$	@Time = (250, 300)
		$\Delta_4 \mathbf{d} = (0.000, -0.700)$	@Time = (350, 400)
Sign change (0.213, -0.245)	$\Delta_1 \mathbf{y} = (0.000, -0.100)$ $\Delta_2 \mathbf{y} = (-0.100, 0.000)$	$\Delta_1 \mathbf{d} = (0.700, 0.000)$	@Time = (50, 100)
		$\Delta_2 \mathbf{d} = (-0.700, 0.000)$	@Time = (150, 200)
		$\Delta_3 \mathbf{d} = (0.000, 0.700)$	@Time = (250, 300)
		$\Delta_4 \mathbf{d} = (0.000, -0.300)$	@Time = (350, 400)

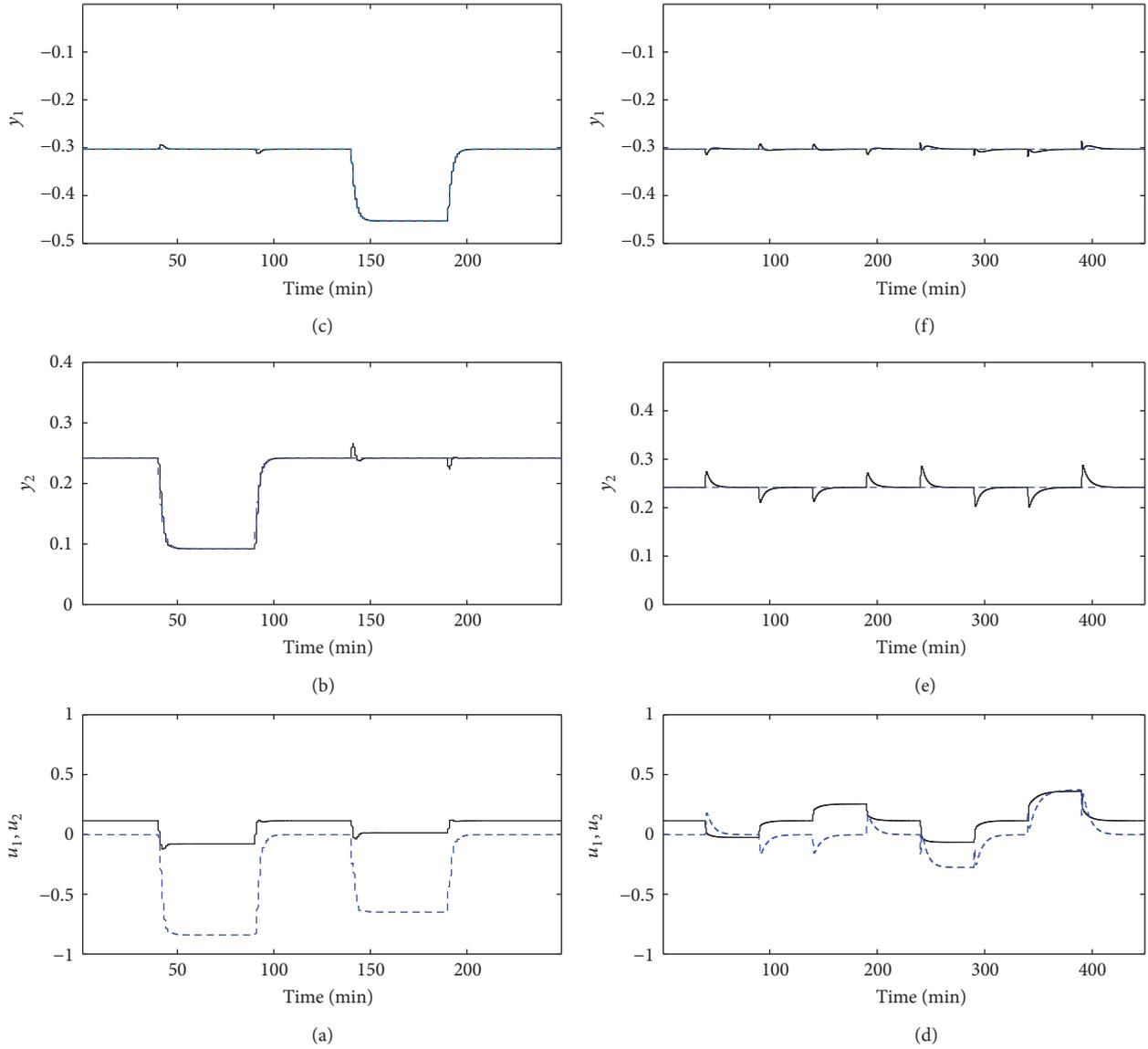


FIGURE 5: Performance of the gain-scheduling PI in Case 2: (a)–(c) Set-point tracking tests; (d)–(f) disturbance rejection tests. Set-points: dashed blue line; system responses: black line; u_1 : black line; u_2 : blue dashed line.

TABLE 4: Summary of ISE for Case 2.

Conditions	Set-point tracking	Disturbance rejection
Nominal (0.000, 0.000)	$5.83e-2$	$4.96e-2$
Maximum gain (-0.303, 0.242)	$6.5e-3$	$7.78e-2$
Sign change (0.213, -0.245)	$1.97e-3$	$2.08e-2$

neural model. These results have already been published in a previous work [13] for a different model-based controller.

5. Conclusion

In this paper, the use of a gain-scheduling controller strategy based on a dynamic neural network model was presented. This technique was proposed to solve problems concerning the control of highly nonlinear systems, without requiring a controller structure unusual to industrial practice. Starting from a GMC approach, the resulting algorithm is a PI controller, with an adaptive gain based on the neural model.

The performance of the developed control technique was tested in two different cases of nonisothermal CSTRs, in which an exothermic, first-order reaction takes place. The controller was applied to a wide range of set-point changes because in this way the system is forced to operate at very critical conditions, and the robustness of the control system

can be verified. The proposed control scheme was also tested in presence of disturbances in order to demonstrate the capability of the net to properly adjust gain controller values. Good results were obtained, indicating that the proposed algorithm properly adjusts the constant gain value over a wide range of operating conditions. This means that the neural model is able to describe the essential features of the process, and it captures the essential nonlinearities through an effective linear description. This characteristic enhances the adaptive controller robustness, because it performs well in the neighbourhood of the nominal operating conditions without incorporating a linear function in the neural network model [7].

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

Stefania Tronci kindly acknowledges the Fondazione Banco di Sardegna for the financial support.

References

- [1] M. Mulas, S. Tronci, F. Corona et al., "Predictive control of an activated sludge process: an application to the Viikinmäki wastewater treatment plant," *Journal of Process Control*, vol. 35, pp. 89–100, 2015.
- [2] I. Machón-González and H. López-García, "Feedforward nonlinear control using neural gas network," *Complexity*, vol. 2017, Article ID 3125073, 11 pages, 2017.
- [3] G. Cogoni, S. Tronci, R. Baratti, and J. A. Romagnoli, "Controllability of semibatch nonisothermal antisolvent crystallization processes," *Industrial & Engineering Chemistry Research*, vol. 53, no. 17, pp. 7056–7065, 2014.
- [4] J. Na, X. Ren, C. Shang, and Y. Guo, "Adaptive neural network predictive control for nonlinear pure feedback systems with input delay," *Journal of Process Control*, vol. 22, no. 1, pp. 194–206, 2012.
- [5] P. L. Lee and G. R. Sullivan, "Generic model control (GMC)," *Computers & Chemical Engineering*, vol. 12, no. 6, pp. 573–580, 1988.
- [6] B. J. Cott and S. Macchietto, "Temperature control of exothermic batch reactors using generic model control," *Industrial & Engineering Chemistry Research*, vol. 28, no. 8, pp. 1177–1184, 1989.
- [7] T. D. Knapp, H. M. Budman, and G. Broderick, "Adaptive control of a CSTR with a neural network model," *Journal of Process Control*, vol. 11, no. 1, pp. 53–68, 2001.
- [8] A. Vega, F. Diez, and J. M. Alvarez, "Programmed cooling control of a batch crystallizer," *Computers & Chemical Engineering*, vol. 19, pp. 471–476, 1995.
- [9] E. E. Ekpo and I. M. Mujtaba, "Evaluation of neural networks-based controllers in batch polymerisation of methyl methacrylate," *Neurocomputing*, vol. 71, no. 7–9, pp. 1401–1412, 2008.
- [10] R. Kamesh, P. S. Reddy, and K. Y. Rani, "Comparative study of different cascade control configurations for a multiproduct semibatch polymerization reactor," *Industrial & Engineering Chemistry Research*, vol. 53, no. 38, pp. 14735–14754, 2014.
- [11] S. M. Alsadaie and I. M. Mujtaba, "Generic model control (GMC) in multistage flash (MSF) desalination," *Journal of Process Control*, vol. 44, pp. 92–105, 2016.
- [12] C. Foscoliano, S. Del Vigo, M. Mulas, and S. Tronci, "Predictive control of an activated sludge process for long term operation," *Chemical Engineering Journal*, vol. 304, pp. 1031–1044, 2016.
- [13] R. Baratti, A. Servida, and S. Tronci, "Neural DMC control strategy for a CSTR in presence of noise," in *IFAC Symposium Dycops-6*, G. Stephanopoulos, Ed., vol. 34, pp. 680–694, 6th edition, 2001.
- [14] S. S. Ge, C. C. Hang, and T. Zhang, "Nonlinear adaptive control using neural networks and its application to CSTR systems," *Journal of Process Control*, vol. 9, no. 4, pp. 313–323, 1999.
- [15] G. M. Scott and W. H. Ray, "Creating efficient nonlinear neural network process models that allow model interpretation," *Journal of Process Control*, vol. 3, no. 3, pp. 163–178, 1993.
- [16] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals through Simulations*, vol. 672, Wiley, New York, NY, USA, 2000.
- [17] B. A. Ogunnaike and W. H. Ray, *Process Dynamics, Modeling and Control*, Oxford University Press, 1994.
- [18] G. Lightbody and G. W. Irwin, "Direct neural model reference adaptive control," *IEE Proceedings Control Theory and Applications*, vol. 142, no. 1, pp. 31–43, 1995.

Research Article

Modeling and Error Compensation of Robotic Articulated Arm Coordinate Measuring Machines Using BP Neural Network

Guanbin Gao,¹ Hongwei Zhang,¹ Hongjun San,¹ Xing Wu,¹ and Wen Wang²

¹Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming, China

²School of Mechanical Engineering, Hangzhou Dianzi University, Hangzhou 310018, China

Correspondence should be addressed to Guanbin Gao; gbgao@163.com

Received 14 July 2017; Accepted 17 September 2017; Published 18 October 2017

Academic Editor: Guang Li

Copyright © 2017 Guanbin Gao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Articulated arm coordinate measuring machine (AACMM) is a specific robotic structural instrument, which uses D-H method for the purpose of kinematic modeling and error compensation. However, it is difficult for the existing error compensation models to describe various factors, which affects the accuracy of AACMM. In this paper, a modeling and error compensation method for AACMM is proposed based on BP Neural Networks. According to the available measurements, the poses of the AACMM are used as the input, and the coordinates of the probe are used as the output of neural network. To avoid tedious training and improve the training efficiency and prediction accuracy, a data acquisition strategy is developed according to the actual measurement behavior in the joint space. A neural network model is proposed and analyzed by using the data generated via Monte-Carlo method in simulations. The structure and parameter settings of neural network are optimized to improve the prediction accuracy and training speed. Experimental studies have been conducted to verify the proposed algorithm with neural network compensation, which shows that 97% error of the AACMM can be eliminated after compensation. These experimental results have revealed the effectiveness of the proposed modeling and compensation method for AACMM.

1. Introduction

The articulated arm coordinate measuring machine (AACMM) is a specific precision measurement instrument, which has a mechanical structure similar to a robotic arm [1]. As an open-linkage structure, the linkages of AACMM are connected by joints one after another, which results in error propagation and accumulation [2, 3]. Therefore, the accuracy of AACMM is relatively lower than that of normal coordinate measuring machines [4, 5].

There are mainly two ways to improve the accuracy of AACMM [6–8]. The first solution is to improve the hardware accuracy of the AACMM, such as the use of high-precision angle encoders and improvement of the accuracy of manufacturing and assembly. However, improving the accuracy of the hardware will inevitably lead to increased costs. Furthermore, the improvement on the accuracy of hardware is also technically restricted [9]. The second solution is to conduct calibration which improves the accuracy

of the AACMM by identifying the errors of the parameters and then compensating them in the kinematic models [10]. Currently, the parameter identification and compensation are the major strategies to improve the accuracy of AACMM, and they have been the current topic in this field [11, 12].

Santolaria et al. [13] presented an identification method for an AACMM using nominal data collected by a ball bar gauge, along with the algorithm and objective functions. The method is based on a new approach including the terms regarding the measurement accuracy and repeatability. Zheng et al. [14] proposed multiple measurement models built by Denavit-Hartenberg's (DH) notation, where the homemade standard rod components are used as a calibration tool and the Levenberg-Marquardt algorithm is applied to solve the structural parameters in the measurement models. Benciolini and Vitti [1] presented a mathematical model of a kinematic chain used in the design and implementation of the algorithms that are necessary for the operation and the

identification of an AACMM. Gao et al. [6] presented a self-calibration method based on a D-H model and constrained movement chains.

Currently, most of the above-mentioned kinematic models of AACMM have been developed based on D-H method, which uses structural parameters to describe the relationship between the coordinates of the probe and the rotation angles of the joints. However, the real structural parameters cannot be obtained due to the complex couplings of the parameters in the D-H model [2]. Moreover, the structural parameters are not stable due to the measuring force, gravity, environmental temperature, and so on. Therefore, D-H method is not complete in the modeling and compensation of AACMMs. Motivated by these facts, a new modeling and error compensation method for AACMMs with BP neural network (BPNN) will be proposed in this paper.

First, the kinematic model of AACMMs with BPNN is established with the joint angles being the input and the coordinates of the probe being the output. Then, simulations were conducted to verify the feasibility of the proposed model. The training data was generated by Mont-Carlo method. And the setting parameters of the model were optimized based on the analysis of the further prediction simulations. To avoid useless training and improve the training efficiency and prediction accuracy, a data acquisition strategy was developed according to the actual measurement behavior in the joint space. To facilitate practical implementations, experimental studies have been conducted based on a practical AACMM. These experimental results have indicated the effectiveness of the proposed modeling and error compensation method.

In comparison to some other modeling and error compensation methods for AACMM, the idea proposed in this paper does not need complex kinematics calculations and structural parameter identification. Moreover, it can also compensate for the effect of environment temperature variation, measuring force, gravity, and other factors affecting the measurement accuracy.

The paper is organized as follows: Section 2 presents the kinematic modeling with D-H method and BPNN, respectively; simulations are conducted to determine the structure of the BPNN and optimize the setting parameters in Section 3 and experimental results are given in Section 4. Section 5 provides conclusions.

2. Kinematic Modeling of AACMM

2.1. Kinematic Modeling Based on D-H Method. According to D-H method [15], we built the coordinate systems for each joint of the AACMM, as shown in Figures 1 and 2.

As shown in Figure 1, the coordinates (x, y, z) of the probe can be derived through six homogeneous transformations as follows:

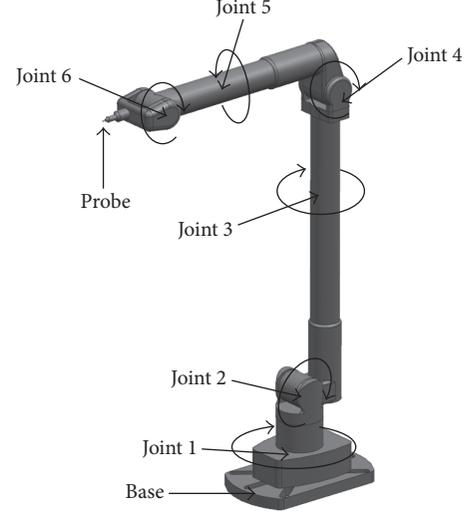


FIGURE 1: The structure of the AACMM.

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = T_{0,1} \cdot T_{1,2} \cdot T_{2,3} \cdot T_{3,4} \cdot T_{4,5} \cdot T_{5,6} \cdot \begin{bmatrix} 0 \\ 0 \\ l \\ 1 \end{bmatrix} \\ = \prod_{i=1}^6 \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cos \alpha_i & \sin \theta_i \sin \alpha_i & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cos \alpha_i & -\cos \theta_i \sin \alpha_i & a_i \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) \\ \cdot \begin{bmatrix} 0 \\ 0 \\ l \\ 1 \end{bmatrix},$$

where d_i is the linkage length, a_i is the joint length, α_i is the torsion angle, and θ_i is the joint angle. The nominal value of each parameter in (1) is shown in Table 1.

2.2. Kinematic Modeling with BP Neural Network. There are various factors that can influence the movement uncertainty of AACMMs, for example, error of structural parameters, deformation caused by the measuring force and gravity, rotation error of joints, and thermal deformation. Hence, it is difficult to model and compensate for all these factors explicitly.

Back-Propagation Neural Network (BPNN), originally proposed by Zipser et al. [16, 17], is one of the most widely used artificial neural network with a powerful ability of nonlinear mapping and self-learning [18–20]. With this ability, BPNN can predict the errors of AACMMs caused by various factors, which makes it possible to compensate for the comprehensive errors and improve the movement uncertainty of AACMM. Hence, this paper will apply BPNN

TABLE I: The nominal structural parameters of the AACMM.

Linkage number i	a_i [mm]	d_i [mm]	$\Delta\theta_i$ [°]	α_i [°]	l [mm]
1	0	376	0	-90	98
2	62	0	0	-90	
3	0	751	0	-90	
4	62	0	0	-90	
5	0	500	0	-90	
6	0	15	0	90	

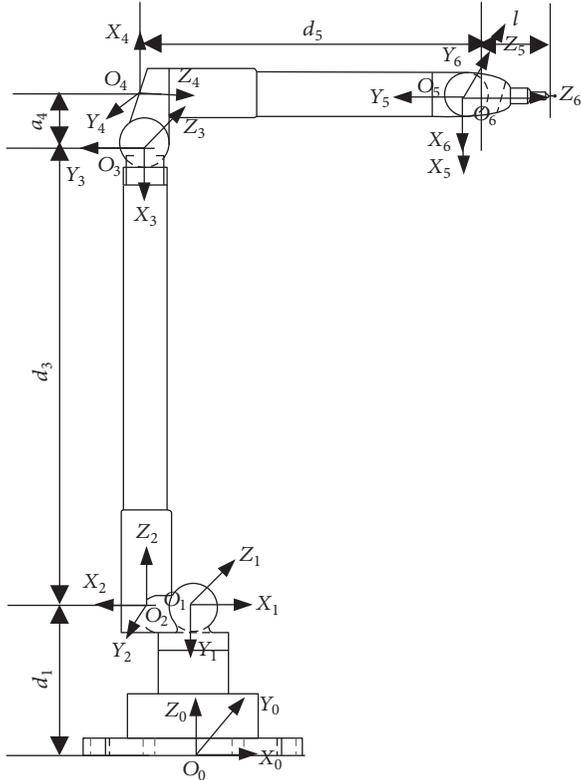


FIGURE 2: Coordinate systems of the AACMM.

to model AACMMs and further improve the modeling accuracy.

The kinematic model of AACMMs based on BPNN is established as shown in (2) with joint angles being the input and the position (coordinates) of the probe being the output.

$$P_j(\boldsymbol{\theta}) = f\left(\sum_{j=1}^n w_{ij} x_j(\boldsymbol{\theta}) - T_j\right), \quad (2)$$

where x_j is the input (six joint angles), w_{ij} is the neural network's weight from neurons j to i , T_j is the threshold of the neuron j , f is the activation function of neurons, and $P_j(\boldsymbol{\theta})$ is the position of the probe.

The structure of the kinematic model based on BPNN is shown in Figure 3. There are six neurons in the input layer which are the six joint angles ($\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6$). The three neurons in the output layer are the coordinates (x, y, z) of the probe. The number of the hidden layers and the number

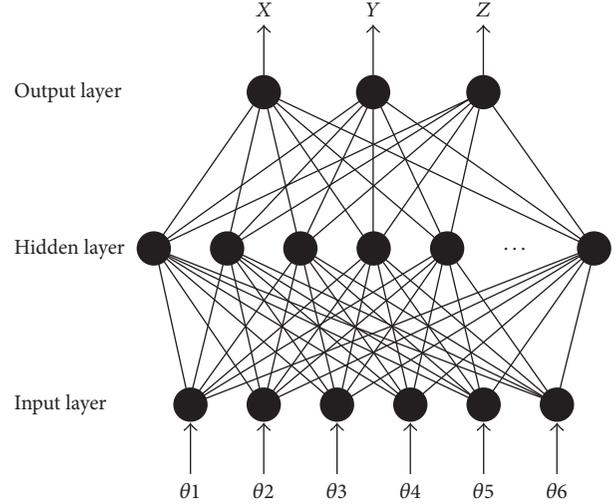


FIGURE 3: The structure of the kinematic model based on BPNN.

of the neurons in each hidden layer will be determined by using optimization methods in Sections 3 and 4. The neurons between the layers are all fully connected.

Theoretically, it is shown that the model of neural network (2) can be used to simulate the AACMM model (1) with arbitrarily accuracy and hence can compensate for the modeling errors. However, in practice, we need to determine the parameters of neural network (e.g., number of neurons n and weights w_{ij} of the neural network). In this paper, we will first determine the parameters of the neural network via simulations, where the BP algorithm will be used to obtain the optimized weights of neural network. Then, practical experiments based on an AACMM will be given to validate the improved modeling accuracy.

3. Determining the Structure and Parameters of the BPNN

To validate the proposed kinematic model based on BPNN and obtain the optimized parameters, Monte-Carlo simulations are carried out to generate training data set for neural network learning.

3.1. Acquisition of Training Data. According to (2), it is known that the input of the model is the six joint angles of the AACMM and the output is the coordinates of the probe. The pose of the AACMM is determined by the six joint angles

TABLE 2: Simulation results of different parameters of BPNN for the AACMM.

Simulation number	TD	HL	NN	LR	GA	Epoch [times]	Training time [seconds]	Maximum error [mm]	Average error [mm]
(1)	10000	1	10	0.1	$4e-5$	1000	124	25.9080	8.511
(2)	8000	1	10	0.1	$4e-5$	516	51	20.4858	8.6132
(3)	5000	1	10	0.1	$4e-5$	1000	64	22.6937	9.4585
(4)	20000	1	10	0.1	$4e-5$	586	253	22.3574	8.9189
(5)	9000	1	10	0.1	$4e-5$	1000	115	20.6444	8.3117
(6)	10000	1	10	0.1	$1e-5$	1000	127	26.887	8.8126
(7)	10000	1	10	0.1	$4e-6$	1000	126	25.8964	8.1825
(8)	10000	1	10	1	$4e-5$	360	123	21.5603	8.7369
(9)	10000	1	10	0.01	$4e-5$	1000	125	18.1312	8.1551
(10)	10000	1	10	0.001	$4e-5$	1000	125	27.6223	8.8921
(11)	10000	1	10	10	$4e-5$	136	17	31.2093	10.1348
(12)	10000	1	3	0.1	$4e-5$	1000	93	90.7123	34.1409
(13)	10000	1	4	0.1	$4e-5$	293	25	68.0939	26.4297
(14)	10000	1	13	0.1	$4e-5$	589	89	14.8641	6.3235
(15)	10000	1	50	0.1	$4e-5$	40	21	6.1157	2.5537
(16)	10000	1	100	0.1	$4e-5$	17	12	9.4408	2.8841
(17)	10000	2	[20, 20]	0.1	$4e-5$	38	23	11.5531	2.4639

which can be generated with the Mont-Carlo method [21]. Then the coordinates of the probe can be calculated by (1). If there are no errors with the structural parameters, the coordinates of the probe would be the theoretic value without any bias. The joint angles and the coordinates will be used for training the BPNN model.

3.2. Structure of the BPNN. The performance of BPNN is affected by many parameters such as the amount of training data (TD), number of hidden layers (HL), number of neuron nodes (NN), leaning rate (LR), goal accuracy (GA), and active function (AF) [22, 23]. To obtain the optimal parameters, extensive simulations were conducted, where BP algorithm will be used to obtain the optimized neural network's weights. The maximum epochs are set to be 1000. Tansig and Marquardt are selected as the active function and training function, respectively. For the high nonlinearities and complex dimension of the BPNN model built for the AACMM, there is some randomness in the result of the simulation. The results of the simulations were shown as Table 2.

To test the training results, another 10 groups of joint angles were used for prediction. The prediction and its error response of the simulation of number (1) are shown in Figure 4. As it is shown, after training the BPNN can predict the coordinates (x, y, z) of the probe of the AACMM with fairly good error response.

From the simulations of numbers (1)–(5), it is known that increasing the amount of the training data can improve the prediction precision, while the speed of error convergence will be reduced. Furthermore, the influence of increasing the number of the training data is not obvious after it reaches a certain amount. The simulation reveals that 9000 groups of training data are appropriate in our case study. Moreover,

the choice of the neural network's learning rate affects the stability and convergence of BPNN model directly. In general, a large learning rate can speed up the convergence rate and reduce training time but will lead to instability of the BPNN. A small learning rate will increase the training time greatly. To balance the prediction precision and the training time, the learning rate of the BPNN is set to 0.001 according to the results of simulations numbers (1) and (8)–(11). Simulations of numbers (12)–(16) are used to determine the number of nodes in the hidden neurons, among which the parameters of numbers (12)–(14) are determined by empirical formulas. According to these results, the number of neurons in the single hidden layer is set to 50. Simulation of number (17) is based on a double hidden layer BPNN; comparing with number (15) we know that the double hidden layer BPNN can satisfy the same requirements with fewer epochs and less calculating time than the single BPNN.

3.3. Parameter Optimization of the BPNN. The performance of neural networks is influenced greatly by the parameter settings [24, 25]. To improve the accuracy of BPNN for the AACMM, 12 simulations were conducted to optimize the parameters by taking into account the impact of the couplings from the parameters. To compare the performance of each setting and search for the optimized one, the training time, maximum error, and average error of the AACMM are treated as the objective functions. In the simulations, the number of the training data is 9000, the maximum iteration is 1000, the leaning rate is 0.01, and the target convergence accuracy is set to $4e-9$. The results of the simulations are shown in Table 3.

Table 3 shows that the maximum and average error of the simulation of number (6) is the minimal. Hence, the optimized settings of the BPNN are two HLs (hidden

TABLE 3: Prediction simulation results for optimization.

Simulation number	HL	NN	TF	GA	Epoch [times]	Training time [seconds]	Maximum error [mm]	Average error [mm]
(1)	1	100	LM	$4e-5$	1000	922	2.5544	0.2452
(2)	1	200	LM	$4e-5$	1000	2298	0.6351	0.0967
(3)	1	300	LM	$4e-5$	1000	4161	0.5994	0.0629
(4)	2	[10, 10]	LM	$4e-5$	738	43	17.1128	2.6204
(5)	2	[30, 30]	LM	$4e-5$	1000	1289	0.4888	0.0833
(6)	2	[50, 50]	LM	$4e-5$	582	6431	0.2071	0.0315
(7)	2	[40, 40]	LM	$4e-5$	1000	2177	0.5525	0.0467
(8)	2	[80, 80]	LM	$4e-5$	150	2769	0.7598	0.0497
(9)	2	[50, 50]	LM	$4e-5$	1000	4237	6.4022	0.6042
(10)	2	[50, 50]	LM	$4e-5$	1000	4237	4.3345	0.6983
(11)	2	[50, 50]	ND	$4e-5$	1000	24	633.8425	119.36
(12)	2	[50, 50]	DX	$4e-5$	119	3	383.9885	105.247

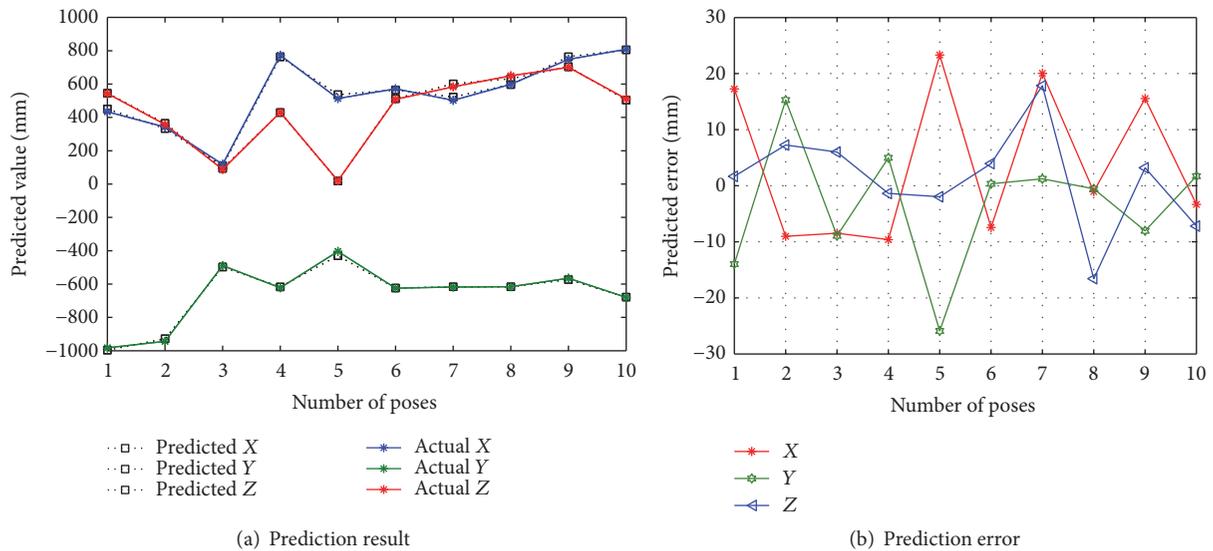


FIGURE 4: The prediction result and prediction error of the simulation number (1).

layers) with neuron nodes (NN) of [50, 50], LM (Levenberg-Marquardt) method as the TF (training function), and hyperbolic tangent S-shaped function as the AF (active function).

4. Practical Experiments

4.1. Error before Compensation. According to (1) and Table 1, the coordinates of the AACMM (shown in Figure 5) were calculated and then were compared with the actual values. The errors of the coordinates are shown in Figure 6, which are mainly due to the errors of structural parameters including the linkage length errors, the torsion angle errors, the joint length errors, the probe length errors, and the joint angle deviations [2]. With these errors, the nominal parameters of the AACMM are not the same as the actual ones and the kinematic model in (1) cannot describe the AACMM system accurately. 200 poses of the AACMM are used for the test. The maximum error is 0.8978 mm and the average error is



FIGURE 5: The articulated arm coordinate measuring machine used in practical experiments.

0.4492 mm, which are not satisfactory for practical use. We can see evident systematic errors in Figure 6 for the reason that there are systematic errors in the nominal structural parameters of Table 1.

TABLE 4: Experimental training results of the BPNN model for AACMM.

Experiment number	HL	NN	LR	GA	Epoch [times]	Training time [seconds]	Maximum error [mm]	Average error [mm]
(1)	2	[50, 50]	0.01	$4e-9$	100	176	0.3199	0.0293
(2)	2	[40, 40]	0.01	$4e-9$	704	625	0.1735	0.0201
(3)	1	300	0.01	$4e-9$	911	1662	0.4059	0.0503
(4)	2	[30, 30]	0.01	$4e-9$	1000	404	0.1428	0.0149

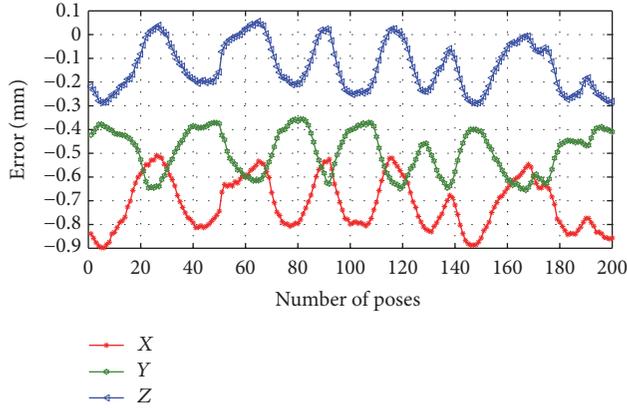


FIGURE 6: The errors of the AACMM before compensation.

4.2. Error Compensation with BPNN Model. To further improve the modeling performance, the proposed neural network model (2) is then adopted, and the neural network's training procedure as proposed in Section 3 will be run again to determine the neural network's parameters and topology. In this case study, 3100 poses of the AACMM are acquired, among which 3000 poses are used to train the BPNN model and the other 100 poses are used to test the prediction accuracy of the BPNN. The training results are shown in Table 4. To simplify the procedure of obtaining the optimum parameters, the training settings in Table 4 are based on the results of Section 3.3.

It is shown that it is feasible to predict the coordinates of the AACMM with satisfactory response by using the prediction values. In particular, from Table 4, we know that the prediction accuracy of the experiment of number (4) is the highest with the maximum error of 0.1428 mm and the average error of 0.0149 mm. Compared to the settings in Section 3.3, the optimum neuron nodes are not the same in simulations and experiments for the difference data used in simulations and experiments. In simulations, the actual noises of the data, for example, variation of temperature, structural deformation of the AACMM, and limited accuracy of the joint angles, are not considered.

After determining the BPNN topology and deriving the BPNN parameters, another 200 groups of poses of the AACMM are acquired to test the trained BPNN model with the settings of number (4) in Table 4. The prediction coordinates and prediction errors of the BPNN model are shown in Figures 7 and 8. The maximum error is 0.0873 mm and the average error is 0.0136 mm. According to the traversal

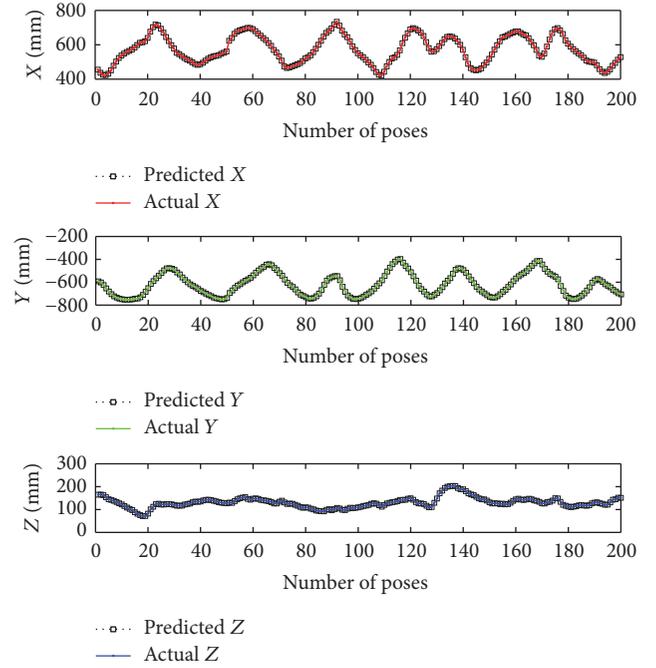


FIGURE 7: The prediction coordinates of the BPNN model for the AACMM.

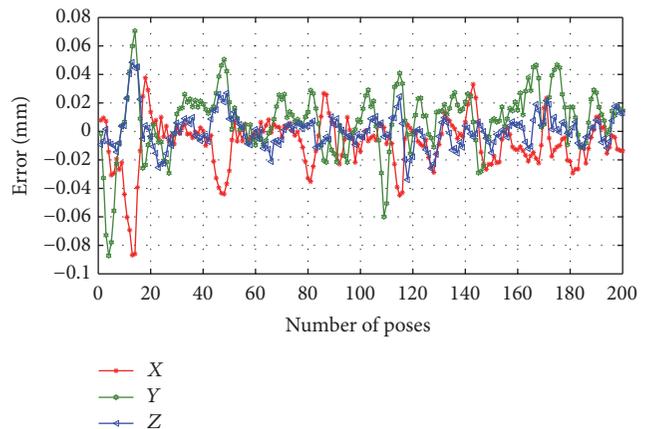


FIGURE 8: Prediction error of the BPNN model for the AACMM.

of the error we found that there are 3 coordinate errors which are greater than 0.08 mm and 13 coordinate errors which are greater than 0.05 mm in the 600 coordinates (200 groups of poses). We have the conclusion that the prediction error is less than 0.05 mm in the principle of 2 Sigma [5]. By comparing

Figures 6 and 8, we know that after the error compensation with the BPNN model the average error of the AACMM is reduced from 0.4492 mm to 0.0136 mm, which means 97% error is removed. Therefore, the BPNN model for AACMM is feasible in practice.

5. Conclusions

A modeling and error compensation approach based on BPNN has been presented for AACMMs in this paper. The kinematic model of AACMMs with BPNN is established with the joint angles being the input and the coordinates of the probe being the output. The training data was first generated by Mont-Carlo method and used to determine the parameters of neural network. And BP training algorithm is used to obtain the optimized neural network's parameters. According to the simulation results, the structure of the BPNN model can be determined with satisfactory error performance. And the setting parameters of the model were optimized based on the analysis of the further prediction simulations. To facilitate practical implementations, experimental studies have been conducted based on a practical AACMM. After error compensation with the BPNN model, 97% average error of the AACMM was eliminated. These experimental results have shown the effectiveness of the proposed modeling and error compensation method. Therefore, for other AACMM applications, the traditional D-H model can be replaced by the BPNN model which avoids using complex kinematics calculation and structural parameter identification. The proposed BPNN model for AACMM can also compensate for the effect of environment temperature, measuring force, gravity, and other factors on the measurement accuracy.

The effectiveness of the proposed modeling and compensation method relies on a training phase which needs to acquire some data set of poses and coordinates of the AACMM. Hence, optimization of the setting of the neural network and use of other new neural networks may further improve the effectiveness of the error compensation. This will be further studied in our future work.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 51465027).

References

- [1] B. Benciolini and A. Vitti, "A new quaternion based kinematic model for the operation and the identification of an articulated arm coordinate measuring machine inspired by the geodetic methodology," *Mechanism and Machine Theory*, vol. 112, pp. 192–204, 2017.
- [2] G. Gao, H. Zhang, X. Wu, and Y. Guo, "Structural Parameter Identification of Articulated Arm Coordinate Measuring Machines," *Mathematical Problems in Engineering*, vol. 2016, Article ID 4063046, 2016.
- [3] L. Yu, H. Zhang, G. Gao, W. Wang, J. Na, and X. Wu, "Kinematic modeling and verification of an articulated arm coordinate measuring machine," in *Proceedings of the Seventh International Symposium on Precision Mechanical Measurements*, Xia'men, China, August, 2015.
- [4] R. Acero, A. Brau, J. Santolaria, and M. Pueo, "Evaluation of a metrology platform for an articulated arm coordinate measuring machine verification under the ASME B89.4.22-2004 and VDI 2617_9-2009 standards," *Journal of Manufacturing Systems*, vol. 42, pp. 57–68, 2017.
- [5] K. Ostrowska, A. Gaška, R. Kupiec, J. Sladek, and K. Gromczak, "Verification of Articulated Arm Coordinate Measuring Machines Accuracy Using LaserTracer System as Standard of Length," *Mapan - Journal of Metrology Society of India*, vol. 31, no. 4, pp. 241–256, 2016.
- [6] G. Gao, J. Na, X. Wu, and Y. Guo, "A self-calibration method for articulated arm coordinate measuring machines," *Transactions of the Canadian Society for Mechanical Engineering*, vol. 40, no. 4, pp. 645–655, 2016.
- [7] X. Feng, C. Xu, J. Zhang, Q. Xu, and Y. Fei, "The establishment and testing of a model called virtual articulated arm coordinate measuring machine," *Chongqing Daxue Xuebao/Journal of Chongqing University*, vol. 39, no. 6, pp. 135–140, 2016.
- [8] Y. Xu and Q. Gao, "AACMM Position accuracy distribution research based on the accuracy of grating encoder," *Chinese Journal of Scientific Instrument*, vol. 38, pp. 336–342, 2017.
- [9] D. Zheng, S. Xie, Z. Luo, J. Zhang, and T. Zhou, "Circular grating eccentricity error modeling and correction of articulated arm CMMs," *Journal of Electronic Measurement and Instrument*, vol. 30, pp. 1568–1574, 2016.
- [10] X. Wang, X. Zhang, L. Chen, H. Yang, and H. Zhang, "Physical design of articulated arm coordinate measuring machine and study on calibration method for parameters of articulated arm," *World Sci-Tech R & D*, vol. 38, pp. 1234–1236, 2016.
- [11] X. Wang, H. Wang, Y. Lu, and P. Zhang, "Parameter calibration method of articulated arm coordinate measuring machine," *Nongye Jixie Xuebao/Transactions of the Chinese Society for Agricultural Machinery*, vol. 47, no. 6, pp. 408–412, 2016.
- [12] Y. Lu, P. Zhang, H. Wang, X. Wang, and C. Zhao, "Modeling of measurement space error of AACMM," in *Proceedings of the 8th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2016*, pp. 416–419, Hangzhou, China, September 2016.
- [13] J. Santolaria, J.-J. Aguilar, J.-A. Yagüe, and J. Pastor, "Kinematic parameter estimation technique for calibration and repeatability improvement of articulated arm coordinate measuring machines," *Precision Engineering*, vol. 32, no. 4, pp. 251–268, 2008.
- [14] D. Zheng, Z. Xiao, and X. Xia, "Multiple measurement models of articulated arm coordinate measuring machines," *Chinese Journal of Mechanical Engineering*, vol. 28, no. 5, pp. 994–998, 2015.
- [15] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Transactions of the ASME*, vol. 22, pp. 215–221, 1955.
- [16] D. Zipser and R. A. Andersen, "A back-propagation programmed network that simulates response properties of a subset of posterior parietal neurons," *Nature*, vol. 331, no. 6158, pp. 679–684, 1988.

- [17] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, pp. 1162–1171, 2017.
- [18] K. Khan and A. Sahai, "A comparison of BA, GA, PSO, BP and LM for training feed forward neural networks in e-learning context," *International Journal of Intelligent Systems and Applications*, vol. 4, no. 7, pp. 23–29, 2012.
- [19] I. A. Basheer and M. Hajmeer, "Artificial neural networks: fundamentals, computing, design, and application," *Journal of Microbiological Methods*, vol. 43, no. 1, pp. 3–31, 2000.
- [20] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.
- [21] M. Duwensee, F. E. Talke, S. Suzuki, J. Lin, and D. Wachsenschwanz, "Direct simulation monte carlo method for the simulation of rarefied gas flow in discrete track recording head/disk interfaces," *Journal of Tribology*, vol. 131, no. 1, pp. 1–7, 2009.
- [22] X. Leng, J. Wang, H. Ji et al., "Prediction of size-fractionated airborne particle-bound metals using MLR, BP-ANN and SVM analyses," *Chemosphere*, vol. 180, pp. 513–522, 2017.
- [23] Z. Li and X. Zhao, "BP artificial neural network based wave front correction for sensor-less free space optics communication," *Optics Communications*, vol. 385, pp. 219–228, 2017.
- [24] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [25] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.

Research Article

Adaptive Neural Network Control for Nonlinear Hydraulic Servo-System with Time-Varying State Constraints

Shu-Min Lu¹ and Dong-Juan Li²

¹College of Science, Liaoning University of Technology, Jinzhou, Liaoning 121001, China

²School of Chemical and Environmental Engineering, Liaoning University of Technology, Jinzhou, Liaoning 121001, China

Correspondence should be addressed to Dong-Juan Li; lidongjuan@live.com

Received 7 July 2017; Accepted 29 August 2017; Published 18 October 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Shu-Min Lu and Dong-Juan Li. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

An adaptive neural network control problem is addressed for a class of nonlinear hydraulic servo-systems with time-varying state constraints. In view of the low precision problem of the traditional hydraulic servo-system which is caused by the tracking errors surpassing appropriate bound, the previous works have shown that the constraint for the system is a good way to solve the low precision problem. Meanwhile, compared with constant constraints, the time-varying state constraints are more general in the actual systems. Therefore, when the states of the system are forced to obey bounded time-varying constraint conditions, the high precision tracking performance of the system can be easily realized. In order to achieve this goal, the time-varying barrier Lyapunov function (TVBLF) is used to prevent the states from violating time-varying constraints. By the backstepping design, the adaptive controller will be obtained. A radial basis function neural network (RBFNN) is used to estimate the uncertainties. Based on analyzing the stability of the hydraulic servo-system, we show that the error signals are bounded in the compacts sets; the time-varying state constraints are never violated and all signals of the hydraulic servo-system are bounded. The simulation and experimental results show that the tracking accuracy of system is improved and the controller has fast tracking ability and strong robustness.

1. Introduction

Throughout history, the hydraulic servo-system has been always widely applied in many aspects, such as aerospace, aviation, navigation, weapon, mining, and metallurgy due to the advantages of high response, high power, large stiffness, strong robustness capability, small volume, and so on. However, serious nonlinear behavior, such as control input saturation [1], state constraint, valve opening, nonlinear friction [2], and model uncertainty [3] (load changes, the parameters variation and the element parameter uncertainty [4] caused by the abrasive, containing external disturbance [5, 6], leakage, and other uncertain nonlinear elements), restricts the development of high-performance closed loop system controller [7, 8].

Recently, in order to solve the uncertain nonlinear problem in the system, the most authors adopted the fuzzy logical system or adaptive neural network (NN) control [9, 10] to approximate the external disturbance [11] and

model uncertainty [12, 13]. In [14, 15], the different stability problems of switched nonlinear systems are solved by using the fuzzy adaptive control. The paper [14] solved the discrete-time switch system tracking control problem, and [15, 16] presented decentralized controller for switched uncertain nonlinear large-scale systems with dead zones based on observer [17]. The article [18] solved the control problem of the nonlinear system with dynamic uncertainties and input dead zone. Similarly, the authors in [19] used the NN to deal with the uncertain problem of nonlinear nonstrict feedback discrete-time systems [20]. The article [21] adopted general projection neural network (GPN) to iteratively solve a quadratic programming (QP) problem, which updates the algorithm of NN to develop the new application. Two neural networks which include the critic NN and the action NN are used to approximate the strategic utility function and to minimize both the strategic utility function and the tracking error in [22], respectively. The articles [23–26] adopt the adaptive NN in quantized nonlinear systems, nonlinear

time-delay systems, and uncertain nonlinear systems. These articles extend the theoretical application of neural network. In [27], an adaptive neural control scheme is presented to take the unknown output hysteresis and computational efficiency into account. The motion/force performances limits of the robotic system will be relaxed. The NN turns out to be a clever theory to solve the uncertain information of the systems from the above articles.

At present, the control problem of hydraulic servo-system has been an active research field by designing the adaptive control technique. The authors in [28] have presented an adaptive controller based on the robust integral of the error signal to deal with modeling uncertainties of the system, and the feedback gain is reduced. In [29], the robust adaptive control ensures uniform boundedness of the thruster assisted position mooring system which is in the transverse direction under the ocean current disturbance. The robust adaptive controller has been designed for single-rod nonlinear hydraulic servo-system where the element parameter uncertainty and nonlinear uncertainty behavior exist in [30]. The article in [31] presents output feedback nonlinear robust control based on an extended state observer (ESO) to solve mismatched modeling uncertainties problem, which is very important for high-accuracy tracking control of hydraulic servo-systems. In [32], considering the influence of friction on the system, an adaptive controller is designed to solve the problem of system friction and parameter uncertainty. In fact, due to the high bearing capacity and high stiffness properties of the hydraulic servo-system, so the state constraints problems are ignored in the environment and the interaction of measurement unit tests process, which lead to too large displacement, velocity, or acceleration causing damage to the measuring equipment during testing process [33]. In tests and experiments, if the initial conditions of the system do not match, it is possible that the displacement, velocity, or acceleration of the system is oversized. At the same time, the constraint for the system is a good way to achieve high accuracy tracking performance. However, none of the above articles considered constraints condition. And considering that the state variables vary with time, the constraint condition should also be considered the factor of time changing. Meanwhile, compared with constant constraints, the time-varying state constraints are more general in the actual systems.

In fact, some systems are often affected by constraints [34], such as temperature constraints in chemical reactions and the speed or acceleration constraints when some mechanical systems suffer physical failure [35]. The authors first present the theory of nonlinear system with output constraints and time-varying output constraints by using Barrier Lyapunov in [36, 37]. However, those articles do not consider the interference of unknown functions. In [38], considering a class of special nonlinear systems with the hysteretic output mechanism and the unmeasured states, adaptive neural output feedback control is presented to guarantee the prescribed convergence of the tracking error [39]. The article in [40] has studied adaptive neural control of uncertain stochastic nonlinear systems with dead zone and output constraint. The continuous stirred tank reactor [41], flexible crane system [42], and robot system [43, 44] all design

adaptive NN controller based on output constraints condition. It is essential that the theories of constraint are combined with the actual systems [45]. Similarly, the state constraints have also been researched by many scientific researchers. For example, the article in [46] presents adaptive fuzzy NN control using impedance learning for a constrained robot. The authors considering time-delay condition investigate the robotic manipulator system [47], nonlinear MIMO unknown systems [48], and common nonlinear systems [49] with full state constraints [50]. The authors further research the nonlinear pure-feedback systems, Nussbaum gain adaptive control, and ABLF adaptive controller to apply them on stochastic nonlinear systems and wheeled mobile robotic system by using the theory of integral barrier Lyapunov function or state constraints in [51], [52], [53], [54], [55], and [56], respectively.

Based on the above descriptions, this article presents an adaptive NN tracking control for hydraulic servo-system based on time-varying state constraint, and hydraulic system adaptive control is first designed with the time-varying state constraints. Finally, when the states of the system are forced to obey bounded time-varying constraint conditions, the high precision tracking performance of the system can be easily realized. Meanwhile, it can be proved that the tracking error signal converges to zero asymptotically and all singles of the hydraulic servo-system are bounded by using the Lyapunov analysis. The experimental results show the availability of the design controller.

2. System Descriptions and Problem Preliminaries

A class of hydraulic servo-system dynamics of the inertial load can be described as the following form:

$$m\ddot{x} = P_e A - \rho\dot{x} - d(x, \dot{x}, t), \quad (1)$$

where x represents the displacement of the inertial load; m is the mass of moving parts; $P_e = P_1 - P_2$, P_1 and P_2 are pressures of the left and right inside the hydraulic cylinder, respectively; A is the corresponding areas of the above-mentioned chamber; ρ represents the viscous friction coefficient; $d(x, \dot{x}, t)$ is defined as the unconsidered disturbance.

Considering the compressibility of the hydraulic oil and ignoring leakage of hydraulic cylinder, the pressure dynamic equation of both chambers is as follows:

$$\frac{V_e}{4B_v} \dot{P}_e = -A\dot{x} - C_m P_e - w + Q, \quad (2)$$

where $V_e = V_{s1} + V_{s2}$ is the total effective cylinder volume; $V_{s1} = V_{I1} + Ax$ and $V_{s2} = V_{I2} - Ax$ represent the total volume of the left and the right chamber inside hydraulic cylinder, respectively; V_{I1} and V_{I2} are the corresponding the initial volume; B_v is the effective bulk modulus of hydraulic cylinder; C_m represents the total internal leakage coefficient of hydraulic cylinder; w represents the modeling error of both chambers; $Q = (Q_1 + Q_2)/2$ is the load-flow of hydraulic cylinder; Q_1 and Q_2 , respectively, represent the supplied oil flow rate to the left chamber and the returned oil flow rate to the right chamber.

Consider that the flow rates Q_1 and Q_2 are the function of the spool displacement x_s . The flow equation of hydraulic cylinder is written as follows:

$$\begin{aligned} Q_1 &= k_f x_s h_1(x_s, P_1), \\ Q_2 &= k_f x_s h_2(x_s, P_2) \end{aligned} \quad (3)$$

with

$$\begin{aligned} h_1(x_s, P_1) &= \begin{cases} (P_s - P_1)^{1/2}, & x_s \geq 0, \\ (P_1 - P_r)^{1/2}, & x_s < 0, \end{cases} \\ h_2(x_s, P_2) &= \begin{cases} (P_2 - P_r)^{1/2}, & x_s \geq 0, \\ (P_s - P_2)^{1/2}, & x_s < 0, \end{cases} \end{aligned} \quad (4)$$

where k_f denotes the flow gain coefficient; P_s is the supply pressure of the oil; P_r represents the return oil pressure.

We assume that the spool displacement x_s is direct proportion with control input voltage u in the situation of ignoring the high frequency; that is, $x_s = k_s u$, $k_s > 0$ is the small gain. Then, (3) can be rewritten as

$$\begin{aligned} Q_1 &= K_f u h_1(u, P_1), \\ Q_2 &= K_f u h_2(u, P_2), \end{aligned} \quad (5)$$

where $K_f = k_f k_s$ represents the total flow gain coefficient. Then, we can conclude that the load-flow of hydraulic cylinder is

$$Q = K_f u (P_s - \text{sgn}(u) P_e)^{1/2}, \quad (6)$$

where $\text{sgn}(u)$ represents the step function as

$$\begin{aligned} \text{sgn}(u) &= \begin{cases} 1, & u \geq 0, \\ -1, & u < 0, \end{cases} \\ P_s &= P_1 + P_2, \\ P_r &= 0, \\ P_e &= P_1 - P_2. \end{aligned} \quad (7)$$

Define state variables $x = [x, \dot{x}, \ddot{x}]^T = [x_1, x_2, x_3]^T$; the dynamics of the hydraulic servo-system (1) can be translated into the following the state space expression based on the pressure dynamic equation (2) and the flow equations (5)-(6):

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= x_3, \\ \dot{x}_3 &= -f_1 x_2 - f_2(x) + f_3 g u, \end{aligned} \quad (8)$$

where

$$\begin{aligned} f_1 &= \frac{4A^2 B_v}{mV_e} - \frac{\rho^2}{m}, \\ f_3 &= \frac{4AB_v K_f}{mV_e}, \\ g &= \sqrt{\frac{(P_s - \text{sgn}(u) P_e)}{2}}, \\ f_2(x) &= \frac{4AB_v C_m P_e}{mV_e} + \frac{4AB_v w}{mV_e} + \frac{\rho A P_e}{m^2} - \frac{\rho}{m} d(x) \\ &\quad + \frac{\dot{d}(x)}{m}. \end{aligned} \quad (9)$$

$f_1(x)$, $f_2(x)$, and $f_3(x)$ are unknown smooth parameter functions. In this paper, the state variable x_i is constrained in the following time-varying compact sets:

$$|x_i(t)| \leq k_{b_i}(t), \quad i = 1, 2, 3, \quad (10)$$

where $k_{b_i}(t)$ represents the time-varying smooth functions.

Assumption 1 (see [37]). There exist the functions $\bar{f}_3(x)$ and $\underline{f}_3(x)$ which satisfy the inequality $0 < \underline{f}_3 \leq f_3 \leq \bar{f}_3$.

Assumption 2 (see [43]). There are functions $Y_0(t)$, $Y_1(t)$, and $Y_2(t)$. We assume that the functions satisfy the inequalities $|Y_{i-1}(t)| \leq |k_{b_i}(t)|$, $i = 1, 2, 3$. Then, the smooth desired trajectory $y_d(t)$ and its time derivatives, respectively, satisfy $|y_d(t)| \leq Y_0(t)$ and $|y_d^{(i)}(t)| \leq Y_{i+1}(t)$, $i = 1, 2$ for all $t > 0$.

Lemma 3 (see [54]). For any time-varying function $k_a(t)$, the following inequality holds for all error function $z(t)$ in the interval $|z(t)| \leq k_a(t)$:

$$\log \frac{k_a^2(t)}{k_a^2(t) - z^2(t)} \leq \frac{z^2(t)}{k_a^2(t) - z^2(t)}. \quad (11)$$

The above-mentioned unknown smooth functions f_1 , $f_2(x)$, and f_3 are approximated by the radial basis function NN as

$$f_i(X) = W_i^T \Phi_i(X), \quad i = 1, 2, 3, \quad (12)$$

where $W_i \in R^{l_i}$ denotes the neural network weight vector and $\Phi_i(X) = [\varphi_{i,1}, \dots, \varphi_{i,l_i}] \in R^{l_i}$ are the basis function vectors with input vector $X \in R^{n_i}$ and the NN node number $l_i > 1$. Gaussian functions $\varphi_{i,j}(X)$ are chosen in the following forms:

$$\begin{aligned} \varphi_{i,j}(X) &= \exp\left(-\frac{\|X - \mu_{i,j}\|^2}{\eta_{i,j}^2}\right), \\ &\quad i = 1, 2, 3, \quad j = 1, \dots, l_i, \end{aligned} \quad (13)$$

where $\mu_{i,j} = [\pi_{i,j,1}, \dots, \pi_{i,j,n_i}]^T$ denotes the center of NN function and $\eta_{i,j}$ is the width of the Gaussian function.

Define the unknown optimal weight vector as the following form:

$$W_i^* = \arg \min_{W_i \in R^i} \left[\sup_{X \in R^{n_i}} |W_i^T \Phi_i(X) - f_i(X)| \right], \quad (14)$$

$$i = 1, 2, 3.$$

Then, there exists the unknown approximation errors $\varepsilon_i(X)$ which has supremum $\bar{\varepsilon}_i$; that is, $|\varepsilon_i(X)| \leq \bar{\varepsilon}_i$. Equation (12) will be rewritten in the approximation equations as follows:

$$f_i(X) = W_i^{*T} \Phi_i(X) + \varepsilon_i(X), \quad i = 1, 2, 3. \quad (15)$$

3. The Adaptive NN Controller Design

Define the track trajectory y_d and the virtual controllers α_1 and α_2 ; we have the track error $z_1 = x_1 - y_d$. Then, let $z_2 = x_2 - \alpha_1$ and $z_3 = x_3 - \alpha_2$. Choose the barrier Lyapunov function candidate as

$$V = \sum_{i=1}^3 V_i + V_N, \quad (16)$$

$$V_i = \frac{1}{2} \log \frac{k_{a_i}^2}{k_{a_i}^2 - z_i^2}, \quad V_N = \frac{1}{2} \sum_{j=1}^3 \widetilde{W}_j^T \Theta_j^{-1} \widetilde{W}_j,$$

where $\log(\cdot)$ represents the natural logarithm; $k_{a_i}(t) = k_{b_i}(t) - D_i(t)$ is boundaries of error vector z_i from subsequent feasibility analysis; Θ_j , $j = 1, 2, 3$ are the positive gain matrices; \widetilde{W}_i is the neural network weight error with $\widetilde{W}_i = \widehat{W}_i - W_i^*$, and note that V is the positive definite and continuously differentiable in the set $|z_i(t)| \leq k_{a_i}(t)$ for $i = 1, 2, 3$.

Based on (16), the time derivative of V is

$$\dot{V} = \sum_{i=1}^3 \dot{V}_i + \dot{V}_N \quad (17)$$

$$= \sum_{i=1}^3 \frac{z_i}{(k_{a_i}^2 - z_i^2)} \left(\dot{z}_i - \frac{\dot{k}_{a_i}}{k_{a_i}} z_i \right) + \sum_{j=1}^3 \widetilde{W}_j^T \Theta_j^{-1} \dot{\widetilde{W}}_j.$$

Design the virtual controllers as

$$\alpha_1 = -(\gamma_1 + \bar{\gamma}_1(t)) z_1 + \dot{y}_d, \quad (18)$$

$$\alpha_2 = -(\gamma_2 + \bar{\gamma}_2(t)) z_2 + \dot{\alpha}_1 - \frac{k_{a_2}^2 - z_2^2}{k_{a_1}^2 - z_1^2} z_1, \quad (19)$$

where the time-varying gains are given by

$$\bar{\gamma}_i(t) = \sqrt{\left(\frac{\dot{k}_{a_i}}{k_{a_i}} \right)^2} + \delta_i, \quad i = 1, 2, 3 \quad (20)$$

for all positive constants δ_i and γ_i . When any \dot{k}_{a_i} is zero, δ_i will ensure that the time derivatives of virtual controllers α_i and controller u are bounded.

Substituting (8) and (18)–(20) into (17), we can obtain

$$\begin{aligned} \dot{V} = & -\sum_{i=1}^2 \left(\gamma_i + \bar{\gamma}_i(t) + \frac{\dot{k}_{a_i}}{k_{a_i}} \right) \frac{z_i^2}{(k_{a_i}^2 - z_i^2)} + \frac{z_2 z_3}{(k_{a_2}^2 - z_2^2)} \\ & + \frac{z_3}{(k_{a_3}^2 - z_3^2)} \left(\dot{x}_3 - \dot{\alpha}_2 - \frac{\dot{k}_{a_3}}{k_{a_3}} z_3 \right) \\ & + \sum_{j=1}^3 \widetilde{W}_j^T \Theta_j^{-1} \dot{\widetilde{W}}_j, \end{aligned} \quad (21)$$

where

$$\begin{aligned} \dot{\alpha}_1 = & \frac{\partial \alpha_1}{\partial x_1} \dot{x}_1 + \sum_{k=0}^1 \left(\frac{\partial \alpha_1}{\partial y_d^{(k)}} y_d^{(k+1)} + \frac{\partial \alpha_1}{\partial k_{a_1}^{(k)}} k_{a_1}^{(k+1)} \right), \\ \dot{\alpha}_2 = & \sum_{j=1}^2 \frac{\partial \alpha_2}{\partial x_j} \dot{x}_j \\ & + \sum_{k=0}^2 \left(\frac{\partial \alpha_2}{\partial y_d^{(k)}} y_d^{(k+1)} + \frac{\partial \alpha_2}{\partial k_{a_1}^{(k)}} k_{a_1}^{(k+1)} + \frac{\partial \alpha_2}{\partial k_{a_2}^{(k)}} k_{a_2}^{(k+1)} \right). \end{aligned} \quad (22)$$

According to approximation equations (15), (21) becomes

$$\begin{aligned} \dot{V} = & -\sum_{i=1}^2 \left(\gamma_i + \bar{\gamma}_i(t) + \frac{\dot{k}_{a_i}}{k_{a_i}} \right) \frac{z_i^2}{(k_{a_i}^2 - z_i^2)} \\ & + \sum_{j=1}^3 \widetilde{W}_j^T \Theta_j^{-1} \dot{\widetilde{W}}_j + \frac{z_3}{(k_{a_3}^2 - z_3^2)} \left[-W_1^{*T} \Phi_1(X) x_2 \right. \\ & - W_2^{*T} \Phi_2(X) + W_3^{*T} \Phi_3(X) g u + D(x, t) - \dot{\alpha}_2 \\ & \left. + \frac{(k_{a_3}^2 - z_3^2)}{(k_{a_2}^2 - z_2^2)} z_2 - \frac{\dot{k}_{a_3}}{k_{a_3}} z_3 \right], \end{aligned} \quad (23)$$

where $D(x, t)$ represents the systematic disturbance compensation as the following form:

$$D(x, t) = -\varepsilon_1 x_2 - \varepsilon_2 + \varepsilon_3 g u. \quad (24)$$

Choose the controller as follows:

$$\begin{aligned} u = & \frac{1}{\widetilde{W}_3^T \Phi_3 g} \left(-(\gamma_3 + \bar{\gamma}_3(t)) z_3 + \widetilde{W}_1^T \Phi_1 x_2 \right. \\ & \left. + \widetilde{W}_2^T \Phi_2(x) + \dot{\alpha}_2 - \frac{(k_{a_3}^2 - z_3^2)}{(k_{a_2}^2 - z_2^2)} z_2 - \frac{z_3}{2(k_{a_3}^2 - z_3^2)} \right). \end{aligned} \quad (25)$$

Adopting the equations $W_i^* = \widehat{W}_i - \widetilde{W}_i$, $i = 1, 2, 3$ and controller (25), the expression for (23) can be rewritten by the above controller as

$$\begin{aligned} \dot{V} = & -\sum_{i=1}^3 \left(\gamma_i + \bar{\gamma}_i(t) + \frac{\dot{k}_{a_i}}{k_{a_i}} \right) \frac{z_i^2}{(k_{a_i}^2 - z_i^2)} \\ & + \widehat{W}_1^T \left(\Theta_1^{-1} \dot{\widehat{W}}_1 + \Phi_1(X) \frac{x_2 z_3}{(k_{a_3}^2 - z_3^2)} \right) \\ & + \widehat{W}_2^T \left(\Theta_2^{-1} \dot{\widehat{W}}_2 + \Phi_2(X) \frac{z_3}{(k_{a_3}^2 - z_3^2)} \right) \\ & + \widehat{W}_3^T \left(\Theta_3^{-1} \dot{\widehat{W}}_3 - \Phi_3(X) \frac{z_3 g u}{(k_{a_3}^2 - z_3^2)} \right) \\ & + D(x, t) \frac{z_3}{(k_{a_3}^2 - z_3^2)} - \frac{z_3^2}{2(k_{a_3}^2 - z_3^2)^2}. \end{aligned} \quad (26)$$

Based on the above-mentioned definition of $\bar{\gamma}_i(t)$, we can obtain that

$$\bar{\gamma}_i(t) + \frac{\dot{k}_{a_i}}{k_{a_i}} \geq 0, \quad i = 1, 2, 3. \quad (27)$$

Introducing the projection algorithm of adaptive control, we design the adaptive law from (26) as follows:

$$\begin{aligned} \dot{\widehat{W}}_1 = & \begin{cases} -\Theta_1 \left[\Phi_1(X) \frac{x_2 z_3}{(k_{a_3}^2 - z_3^2)} + \sigma_1 \widehat{W}_1 \right], & \Theta_1 \leq 0, \widehat{W}_1 > M_{f_1}, \\ P_{f_1}(\widehat{W}_1, \Phi_1), & \Theta_1 > 0, \widehat{W}_1 = M_{f_1}, \end{cases} \end{aligned} \quad (28)$$

where $P_{f_1}(\widehat{W}_1, \Phi_1)$ represents projection operator as the following form:

$$\begin{aligned} P_{f_1}(\widehat{W}_1, \Phi_1) = & -\Theta_1 \left[\Phi_1(X) \frac{x_2 z_3}{(k_{a_3}^2 - z_3^2)} \right. \\ & \left. - \Phi_1(X) \frac{\widehat{W}_1^T \widehat{W}_1 x_2 z_3}{\|\widehat{W}_1\|^2 (k_{a_3}^2 - z_3^2)} + \sigma_1 \widehat{W}_1 \right]. \end{aligned} \quad (29)$$

M_{f_1} is the minuteness positive constant to ensure the adaptive law $\dot{\widehat{W}}_1 \geq 0$ in any situation.

Similarly, the other adaptive laws can obtain the forms as

$$\dot{\widehat{W}}_2 = \begin{cases} -\Theta_2 \left[\Phi_2(X) \frac{z_3}{(k_{a_3}^2 - z_3^2)} + \sigma_2 \widehat{W}_2 \right], & \Theta_2 \leq 0, \widehat{W}_2 > M_{f_2}, \\ P_{f_2}(\widehat{W}_2, \Phi_2), & \Theta_2 > 0, \widehat{W}_2 = M_{f_2}, \end{cases} \quad (30)$$

$$P_{f_2}(\widehat{W}_2, \Phi_2) = -\Theta_2 \left[\Phi_2(X) \frac{z_3}{(k_{a_3}^2 - z_3^2)} - \Phi_2(X) \frac{\widehat{W}_2^T \widehat{W}_2 z_3}{\|\widehat{W}_2\|^2 (k_{a_3}^2 - z_3^2)} + \sigma_2 \widehat{W}_2 \right], \quad (31)$$

$$\dot{\widehat{W}}_3 = \begin{cases} \Theta_3 \left[\Phi_3(X) \frac{z_3 g u}{(k_{a_3}^2 - z_3^2)} - \sigma_3 \widehat{W}_3 \right], & \Theta_3 \leq 0, \widehat{W}_3 > M_{f_3}, \\ P_{f_3}(\widehat{W}_3, \Phi_3), & \Theta_3 > 0, \widehat{W}_3 = M_{f_3}, \end{cases} \quad (32)$$

$$P_{f_3}(\widehat{W}_3, \Phi_3) = -\Theta_3 \left[\Phi_3(X) \frac{z_3 g u}{(k_{a_3}^2 - z_3^2)} - \Phi_3(X) \frac{\widehat{W}_3^T \widehat{W}_3 z_3 g u}{\|\widehat{W}_3\|^2 (k_{a_3}^2 - z_3^2)} + \sigma_3 \widehat{W}_3 \right]. \quad (33)$$

In particular, when component $\widehat{W}_{3,i}$ of \widehat{W}_3 is a small constant $m_{f_{3,i}}$, that is, $\widehat{W}_3 = M_{f_3}$, we adopt the following component of adaptive law:

$$\dot{\widehat{W}}_{3,i} = \begin{cases} -\Theta_{3,i} \left[\Phi_{3,i}(X) \frac{z_3 g u}{(k_{a_3}^2 - z_3^2)} + \sigma_3 \widehat{W}_{3,i} \right], \\ 0. \end{cases} \quad (34)$$

Remark 4. Based on the above description (34), we can make the following conclusion. If $\widehat{W}_{3,i} = m_{f_{3,i}}$, we can get $\dot{\widehat{W}}_{3,i} > 0$,

that is, $\widehat{W}_{3,i} \geq m_{f_{3,i}} > 0$ for all any $\widehat{W}_{3,i}$. Then, there does not exist insignificance function for controller u .

Substituting (28), (30), and (32) into (26), we obtain

$$\begin{aligned} \dot{V} \leq & -\sum_{i=1}^3 \left(\gamma_i + \bar{\gamma}_i(t) + \frac{\dot{k}_{a_i}}{k_{a_i}} \right) \frac{z_i^2}{(k_{a_i}^2 - z_i^2)} - \sum_{i=1}^3 \sigma_i \widehat{W}_i^T \widehat{W}_i \\ & + D(x, t) \frac{z_3}{(k_{a_3}^2 - z_3^2)} - \frac{z_3^2}{2(k_{a_3}^2 - z_3^2)^2}. \end{aligned} \quad (35)$$

Using Young's inequality in the last two terms of \dot{V} , we seem to easily get

$$\begin{aligned} -\sigma_i \bar{W}_i^T \bar{W}_i &= -\sigma_i \bar{W}_i^T (\bar{W}_i + W_i^*) \\ &\leq -\sigma_i \|\bar{W}_i\|^2 + \frac{\sigma_i}{2} \|W_i^*\|^2 \\ &\quad + \frac{\sigma_i}{2} \|\bar{W}_i\|^2 \\ &= -\frac{\sigma_i}{2} \|\bar{W}_i\|^2 + \frac{\sigma_i}{2} \|W_i^*\|^2, \end{aligned} \quad (36)$$

$$D(x, t) \frac{z_3}{(k_{a_3}^2 - z_3^2)} \leq \frac{1}{2} D^2(x, t) + \frac{z_3^2}{2(k_{a_3}^2 - z_3^2)^2}.$$

Based on definition (27) and using inequalities (36) and (11) in Lemma 3, (35) can be rewritten as

$$\begin{aligned} \dot{V} \leq & -\sum_{i=1}^3 \gamma_i \log \frac{k_{a_i}^2}{(k_{a_i}^2 - z_i^2)} - \sum_{i=1}^3 \frac{\sigma_i}{2} \|\bar{W}_i\|^2 + \frac{1}{2} D^2(x, t) \\ & + \sum_{i=1}^3 \frac{\sigma_i}{2} \|W_i^*\|^2. \end{aligned} \quad (37)$$

Theorem 5. *A class of hydraulic servo-system is described by (1)–(3) under Assumptions 1 and 2. If the initial displacement $x_1(0)$, the initial velocity $x_2(0)$, and the initial acceleration $x_3(0)$ satisfy $|x_i(0)| \leq k_{b_i}(0)$, $i = 1, 2, 3$, then, the proposed control method has the following properties.*

(1) *The error signals are bounded in the following sets:*

$$z_i(t) \in \Omega_{z_i}, \quad i = 1, 2, 3, \quad (38)$$

where $\Omega_{z_i} := \{z_i \in \mathbb{R} : |z_i(t)| \leq k_{a_i}(t) \sqrt{1 - e^{-V(0)e^{-\psi t} - C/\psi}}\}$, noting that the tracking error signal converges to zero asymptotically.

(2) *The time-varying state constraints are never violated; that is, $|x_i(t)| \leq k_{b_i}(t)$, $\forall t \geq 0, i = 1, 2, 3$.*

(3) *All singles of the hydraulic servo-system are bounded.*

Proof. Please see the Appendix. \square

4. Simulation Example

Simulation has been performed to demonstrate the performance of the proposed approach. The values of system parameters which are used in the simulation are given in Table 1.

The initial states of the hydraulic servo-system are given as $x_1(0) = 0$, $\dot{x}_1(0) = 0$, and $x_1(0) = 0$. The time-varying constraint functions are $k_{b_1}(t) = 2 \sin(5t) + 40$, $k_{b_2}(t) = 2 \sin(5t) + 60$, and $k_{b_3}(t) = 4 \sin(5t) + 150$. The desired trajectory tracking periodic reciprocation of the displacement is given $y_d = 40 \arctan(\sin(0.4\pi t))(1 - e^{-t})$ with $t \in [0, t_f]$ and $t_f = 50$ s. The modeling error disturbance of the

TABLE 1: The system parameters used in the simulation.

Name	Value
m (kg)	2200
P_s (Pa)	20
P_r (Pa)	0
A (m ³)	2×10^{-3}
ρ (Ns/m)	500
V_{11} (m ³)	5×10^{-4}
V_{12} (m ³)	5×10^{-3}
B_v (Pa)	200
C_m (m ⁵ /Ns)	2×10^{-5}
K_f (m ³ /Ns)	8×10^{-2}

both chambers is $w = \sin(t) + 2$. We choose the following controller:

$$\begin{aligned} \alpha_1 &= -(\gamma_1 + \bar{\gamma}_1(t)) z_1 + \dot{y}_d, \\ \alpha_2 &= -(\gamma_2 + \bar{\gamma}_2(t)) z_2 + \dot{\alpha}_1 - \frac{k_{a_2}^2 - z_2^2}{k_{a_1}^2 - z_1^2} z_1, \\ u &= \frac{1}{\bar{W}_3^T \Phi_3 g} \left(-(\gamma_3 + \bar{\gamma}_3(t)) z_3 + \bar{W}_1^T \Phi_1 x_2 + \bar{W}_2^T \Phi_2 \right. \\ &\quad \left. + \dot{\alpha}_2 - \frac{(k_{a_3}^2 - z_3^2)}{(k_{a_2}^2 - z_2^2)} z_2 - \frac{z_3}{2(k_{a_3}^2 - z_3^2)} \right), \end{aligned} \quad (39)$$

where we choose design parameters as $Y_0 = 20$, $Y_1 = 10$, $Y_2 = 50$, $\gamma_1 = 10$, $\gamma_2 = 3$, $\gamma_3 = 6$, $\delta_1 = 0.8$, $\delta_2 = 0.7$, $\delta_3 = 0.6$ the initial weight vector estimation $\bar{W}_1 = -100$, $\bar{W}_2 = 20$, $\bar{W}_3 = 10$, $\Theta_1 = 3$, $\Theta_2 = 2$, $\Theta_3 = 2$, $\sigma_1 = 2$, $\sigma_2 = 0.3$, $\sigma_3 = 0.5$, $\Phi_1 = 0.2$, $\Phi_2 = 0.4$, $\Phi_3 = 0.06$.

Figures 1–6 are illustrated to show the simulation results. Figures 1–3 show the better tracking trajectories performance of the system variable state and minor error curve of the errors on the time-varying constraints, respectively. It can be easily obtained that the time-varying state constraints are not violated. In Figure 4, we can clearly get the conclusion that the tracking errors are all bounded and the tracking error signal curve converges to zero. Figures 5 and 6 show the trajectories of estimation and the controller, respectively. We make the conclusion that the adaptive laws and the controller of the system are all ensured boundedness.

5. Conclusion

An adaptive control algorithm for hydraulic servo-system considering time-varying state constraints has been designed in this paper. The uncertainty and time-varying state constraints problem of the system have been considered in the controller designing process. The uncertainties and the accurate estimation of disturbance have been solved by RBFNN. By designing barrier Lyapunov function to solve the time-varying state constraints problem of system, the stability of the control strategy has been proved. Besides, the proposed method has proved that the error signals are bounded in the

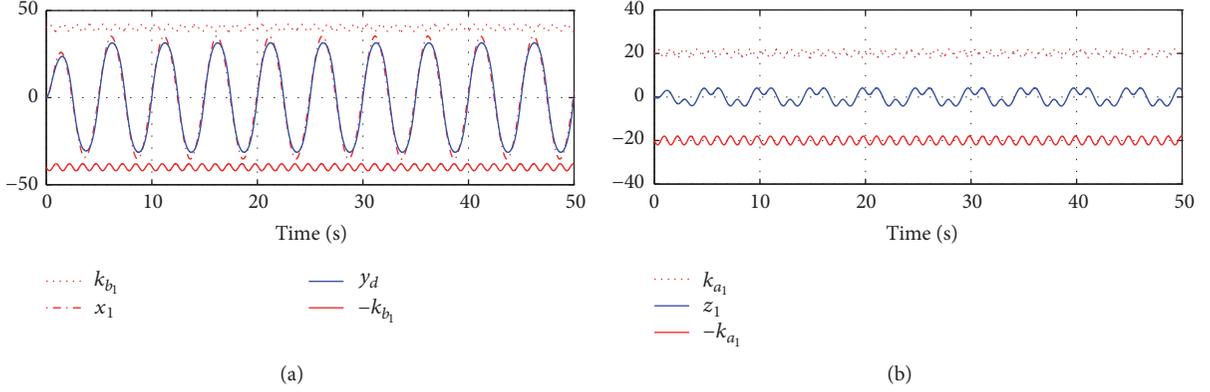


FIGURE 1: (a) Trajectories of x_1 (dashed) and y_d (solid); (b) the trajectory of tracking error z_1 .

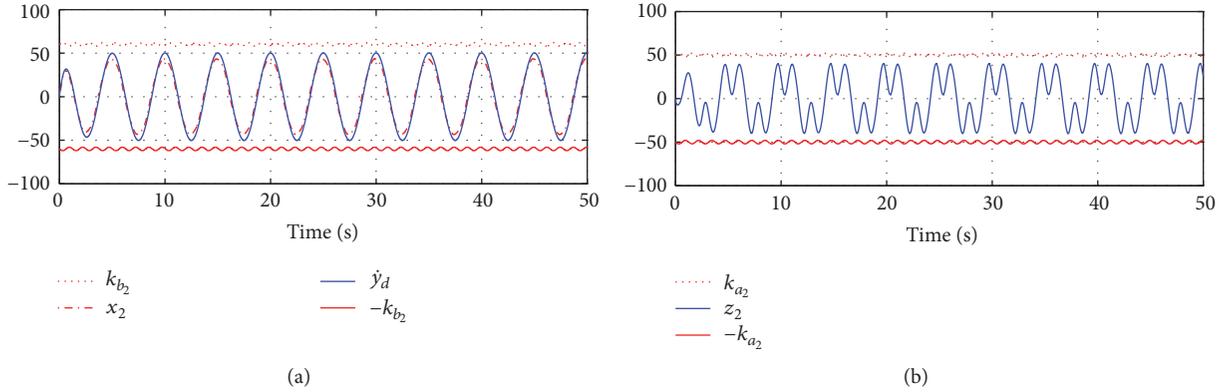


FIGURE 2: (a) Trajectories of x_2 (dashed) and y_d (solid); (b) the trajectory of tracking error z_2 .

small compact sets, noting that the tracking error signal converges to zero asymptotically; the time-varying state constraints are never violated and all signals of the hydraulic servo-system are bounded. The experimental results have showed the effectiveness of the proposed method.

Appendix

Proof of Theorem 5. Based on the state constraints (10) and the definition of errors z_i , the initial error signals are bounded between the sets as $|z_i(0)| \leq k_{a_i}(0)$. From (37), the following can be obtained:

$$\dot{V} \leq -\psi V + C, \quad (\text{A.1})$$

where

$$\psi = \min \{2\gamma_1, 2\gamma_2, 2\}, \quad (\text{A.2})$$

$$C = \frac{1}{2} D^2(x, t) + \sum_{i=1}^3 \frac{\sigma_i}{2} \|W_i^*\|^2.$$

Let us multiply both sides of the above inequality by $e^{\psi t}$; (A.1) can be rewritten as

$$\dot{V} e^{\psi t} + \psi V e^{\psi t} \leq C e^{\psi t}. \quad (\text{A.3})$$

And then we integrate both sides of inequality (A.3) over $[0, t]$ to obtain

$$0 \leq V(t) \leq V(0) e^{-\psi t} + \frac{C}{\psi}. \quad (\text{A.4})$$

Substituting (16) into (A.4), we can get

$$\log \frac{k_{a_i}^2}{k_{a_i}^2 - z_i^2} \leq V(t) \leq V(0) e^{-\psi t} + \frac{C}{\psi}. \quad (\text{A.5})$$

Thus, we have the following inequality based on the transitivity of the inequality from (A.5):

$$\frac{k_{a_i}^2}{k_{a_i}^2 - z_i^2} \leq e^{V(0)e^{-\psi t} + C/\psi}. \quad (\text{A.6})$$

Then, (A.6) can be rewritten as

$$|z_i(t)| \leq k_{a_i}(t) \sqrt{1 - e^{-V(0)e^{-\psi t} - C/\psi}}. \quad (\text{A.7})$$

Furthermore, it can be easily proved that the error signal z_i converges to zero asymptotically based on appropriate parameter design.

Based on Assumption 2, we can get $|y_d(t)| \leq Y_0(t)$ and $|y_d^{(i)}(t)| \leq Y_{i+1}(t)$, $i = 1, 2$. Then, from $|z_1| \leq k_{a_1}$ and

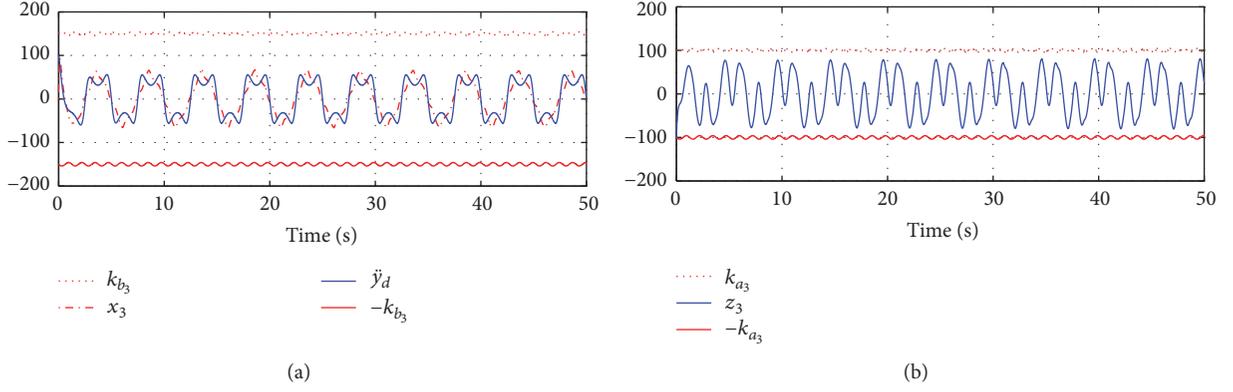


FIGURE 3: (a) Trajectories of x_3 (dashed) and \ddot{y}_d (solid); (b) the trajectory of tracking error z_3 .

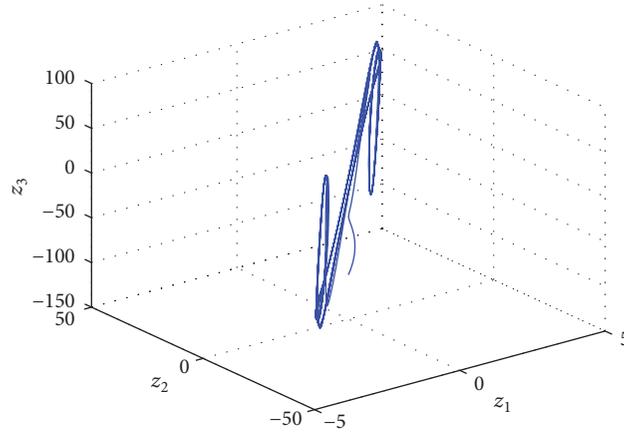


FIGURE 4: The phase portrait of tracking errors z_1 , z_2 , and z_3 .

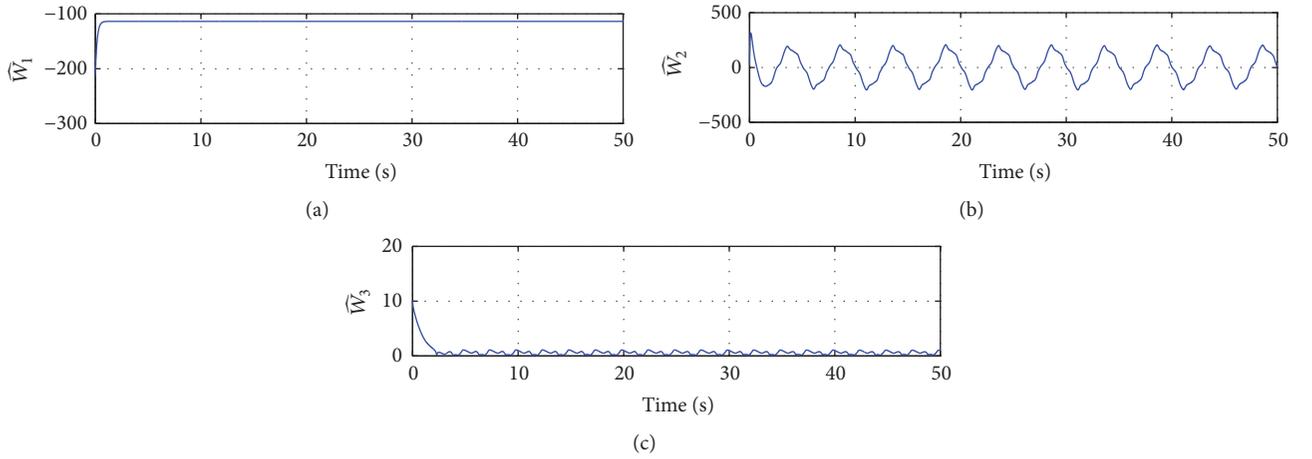


FIGURE 5: (a) The trajectory of \widehat{W}_1 ; (b) the trajectory of \widehat{W}_2 ; (c) the trajectory of \widehat{W}_3 .

$x_1 = z_1 + y_d$, we can get the inequality $|x_1(t)| \leq k_{a_1}(t) + Y_0(t) \leq k_{b_1}(t)$. In (18), the virtual controller α_1 can be bounded from the boundedness of x_1 and \dot{y}_d . It can be seen that there exists the supremum $\bar{\alpha}_1$ of α_1 . From $|z_2| \leq k_{a_2}$ and $x_2 = z_2 + \alpha_1$, it has $|x_2| \leq k_{a_2} + \bar{\alpha}_1 \leq k_{b_2}$. In the same way, we can get the boundedness supremum $\bar{\alpha}_2$ based on the

definition of α_2 in (19). Thus, the state x_3 can be proved by the function k_{b_3} from the above proof of boundedness. So the time-varying state constraints are never violated.

From (A.4) and adaptive laws in (28), (30), and (32), we can get that the neural network weight vector errors \widehat{W}_i are bounded. Because the weight vectors W_i^* are bounded,

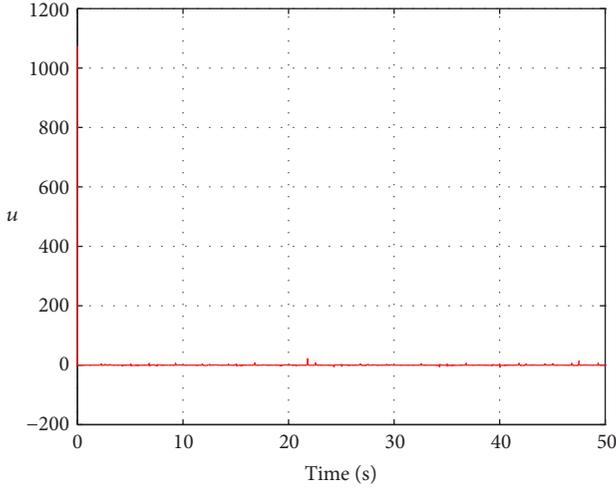


FIGURE 6: The trajectory of u .

the weight vector estimations $\widehat{W}_i = \widetilde{W}_i + W_i^*$ are bounded. Based on the above identified process, the actual controller u consists of the bounded functions $y_d, \dot{y}_d, y_d^{(2)}, y_d^{(3)}, \alpha_1, \alpha_2$, and $z_i, i = 1, 2, 3$. Then, it is easy to obtain that u is bounded. Thus, all the signals of the hydraulic servo-system are bounded.

The proof is completed. \square

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work is supported by the National Natural Science Foundation of China (61603164, 61473139, and 61622303) and the project for Distinguished Professor of Liaoning Province.

References

- [1] W. He, Y. Dong, and C. Sun, "Adaptive neural impedance control of a robotic manipulator with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334–344, 2016.
- [2] L. Chen and Q. Wang, "Adaptive robust control for a class of uncertain MIMO non-affine nonlinear systems," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 1, pp. 105–112, 2016.
- [3] X. Fu, Y. Kang, and P. Li, "Sampled-data observer design for a class of stochastic nonlinear systems based on the approximate discrete-time models," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 507–511, 2017.
- [4] W. He, S. S. Ge, B. V. How, Y. S. Choo, and K.-S. Hong, "Robust adaptive boundary control of a flexible marine riser with vessel dynamics," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 47, no. 4, pp. 722–732, 2011.
- [5] H. C. Lu and W. C. Lin, "Robust controller with disturbance rejection for hydraulic servo systems," *IEEE Transactions on Industrial Electronics*, vol. 40, no. 1, pp. 157–162, 1993.
- [6] M. Chen, "Robust tracking control for self-balancing mobile robots using disturbance observer," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 458–465, 2017.
- [7] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, 2017.
- [8] W. He, S. Nie, T. Meng, and Y.-J. Liu, "Modeling and vibration control for a moving beam with application in a drilling riser," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1036–1043, 2017.
- [9] M. Yue, L. J. Wang, and T. Ma, "Neural network based terminal sliding mode control for WMRs affected by an augmented ground friction with slippage effect," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 3, pp. 498–506, 2017.
- [10] H. Wang, P. X. Liu, S. Li, and D. Wang, "Adaptive neural output-feedback control for a class of nonlower triangular nonlinear systems with unmodeled dynamics," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–11.
- [11] N. Aguila-Camacho and M. A. Duarte-Mermoud, "Improving the control energy in model reference adaptive controllers using fractional adaptive laws," *IEEE/CAA Journal of Automatica Sinica*, vol. 3, no. 3, pp. 332–337, 2016.
- [12] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, 2017.
- [13] B. Xu, Z. Shi, C. Yang, and F. Sun, "Composite neural dynamic surface control of a class of uncertain nonlinear systems in strict-feedback form," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2626–2634, 2014.
- [14] H. Wang, Z. Wang, Y.-J. Liu, and S. Tong, "Fuzzy tracking adaptive control of discrete-time switched nonlinear systems," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 316, pp. 35–48, 2017.
- [15] S. Tong, L. Zhang, and Y. Li, "Observed-based adaptive fuzzy decentralized tracking control for switched uncertain nonlinear large-scale systems with dead zones," *IEEE Transactions on Systems, Man and Cybernetics: Systems*, vol. 46, no. 1, pp. 37–47, 2015.
- [16] M. Chen and G. Tao, "Adaptive fault-tolerant control of uncertain nonlinear large-scale systems with unknown dead zone," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1851–1862, 2016.
- [17] S. Sui, Y. Li, and S. Tong, "Observer-based adaptive fuzzy control for switched stochastic nonlinear systems with partial tracking errors constrained," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 12, pp. 1605–1617, 2016.
- [18] H. Wang, H. R. Karimi, P. X. Liu, and H. Yang, "Adaptive neural control of nonlinear systems with unknown control directions and input dead-zone," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–11.
- [19] Y.-J. Liu, S. Li, S. Tong, and C. L. Chen, "Neural approximation-based adaptive control for a class of nonlinear nonstrict feedback discrete-time systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1531–1541, 2017.
- [20] Z. Wang, L. Liu, H. Zhang, and G. Xiao, "Fault-tolerant controller design for a class of nonlinear MIMO discrete-time systems via online reinforcement learning algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 5, pp. 611–622, 2016.
- [21] Z. Li, H. Xiao, C. Yang, and Y. Zhao, "Model predictive control of nonholonomic chained systems using general projection neural networks optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 10, pp. 1313–1321, 2015.

- [22] B. Xu, C. Yang, and Z. Shi, "Reinforcement learning output feedback NN control using deterministic learning technique," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 635–641, 2014.
- [23] F. Wang, B. Chen, C. Lin, J. Zhang, and X. Meng, "Adaptive neural network finite-time output feedback control of quantized nonlinear systems," *IEEE Transactions on Cybernetics*, pp. 1–10.
- [24] H. Wang, P. X. Liu, and S. Liu, "Adaptive neural synchronization control for bilateral teleoperation systems with time delay and backlash-like hysteresis," *IEEE Transactions on Cybernetics*, 2017.
- [25] M. Chen and S. Ge, "Adaptive neural output feedback control of uncertain nonlinear systems with unknown hysteresis using disturbance observer," *IEEE Transactions on Industrial Electronics*, 2015.
- [26] F. Wang, B. Chen, C. Lin, and X. Li, "Distributed adaptive neural control for stochastic nonlinear multiagent systems," *IEEE Transactions on Cybernetics*, vol. 47, no. 7, pp. 1795–1803, 2017.
- [27] Z. Liu, C. Chen, Y. Zhang, and C. L. P. Chen, "Adaptive neural control for dual-arm coordination of humanoid robot with unknown nonlinearities in output mechanism," *IEEE Transactions on Cybernetics*, vol. 45, no. 3, pp. 521–532, 2015.
- [28] J. y. Yao, Z. x. Jiao, D. w. Ma, and L. Yan, "High-accuracy tracking control of hydraulic rotary actuators with modelling uncertainties," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 2, pp. 633–641, 2014.
- [29] W. He, S. Zhang, and S. S. Ge, "Robust adaptive control of a thruster assisted position mooring system," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 50, no. 7, pp. 1843–1851, 2014.
- [30] B. Yao, F. p. Bu, J. Reedy, and C. T. C. Chiu, "Adaptive robust motion control of single-rod hydraulic actuators: theory and experiments," *IEEE/ASME Transactions on Mechatronics*, vol. 5, no. 1, pp. 79–91, 2000.
- [31] J. Y. Yao, Z. X. Jiao, and D. W. Ma, "Extended-state-observer-based output feedback nonlinear robust control of hydraulic systems with backstepping," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 6285–6293, 2014.
- [32] J. Yao, W. Deng, and Z. Jiao, "Adaptive control of hydraulic actuators with LuGre model-based friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 10, pp. 6469–6477, 2015.
- [33] S. Tong, J. Fang, and Y. Zhang, "Output tracking control of a hydrogen-air PEM fuel cell," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 2, pp. 273–279, 2017.
- [34] Z. Fu, W. Xie, S. Rakheja, and J. Na, "Observer-based adaptive optimal control for unknown singularly perturbed nonlinear systems with input constraints," *IEEE/CAA Journal of Automatica Sinica*, vol. 4, no. 1, pp. 48–57, 2017.
- [35] Z. J. Li, S. T. Xiao, S. Z. S. Ge, and H. Su, "Constrained multi-legged robot system modeling and fuzzy control with uncertain kinematics and dynamics incorporating foot force optimization," *IEEE Transactions on Systems, Man, & Cybernetics: Systems*, vol. 46, no. 1, pp. 1–15, 2016.
- [36] K. P. Tee, S. S. Ge, and E. H. Tay, "Barrier Lyapunov functions for the control of output-constrained nonlinear systems," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 45, no. 4, pp. 918–927, 2009.
- [37] K. P. Tee, B. Ren, and S. S. Ge, "Control of nonlinear systems with time-varying output constraints," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 47, no. 11, pp. 2511–2516, 2011.
- [38] Z. Liu, G. Lai, Y. Zhang, and C. L. Chen, "Adaptive neural output feedback control of output-constrained nonlinear systems with unknown output nonlinearity," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 8, pp. 1789–1802, 2015.
- [39] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 6, pp. 740–749, 2016.
- [40] H. Li, L. Bai, L. Wang, Q. Zhou, and H. Wang, "Adaptive neural control of uncertain nonstrict-feedback stochastic nonlinear systems with output constraint and unknown dead zone," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2048–2059, 2017.
- [41] D.-J. Li and D.-P. Li, "Adaptive controller design-based neural networks for output constraint continuous stirred tank reactor," *Neurocomputing*, vol. 153, pp. 159–163, 2015.
- [42] W. He, S. Zhang, and S. S. Ge, "Adaptive control of a flexible crane system with the boundary output constraint," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 8, pp. 4126–4133, 2014.
- [43] Y. J. Liu, S. M. Lu, and S. C. Tong, "Neural network controller design for an uncertain robot with time-varying output constraint," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2060–2068, 2017.
- [44] Z. Li, Q. Ge, W. Ye, and P. Yuan, "Dynamic balance optimization and control of quadruped robot systems with flexible joints," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 10, pp. 1338–1351, 2016.
- [45] M. Chen, "Constrained control allocation for overactuated aircraft using a neurodynamic model," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 12, pp. 1630–1641, 2016.
- [46] W. He and Y. Dong, "Adaptive fuzzy neural network control for a constrained robot using impedance learning," *IEEE Transactions on Neural Networks and Learning Systems*, no. 99, pp. 1–13, 2017.
- [47] D. P. Li and D. J. Li, "Adaptive neural tracking control for an uncertain state constrained robotic manipulator with time-varying delays," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [48] D. P. Li, D. J. Li, Y. J. Liu, S. Tong, and C. L. P. Chen, "Approximation-based adaptive neural tracking control of nonlinear mimo unknown time-varying delay systems with full state constraints," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3100–3109, 2017.
- [49] D. Li and D. Li, "Adaptive neural tracking control for nonlinear time-delay systems with full state constraints," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1590–1601, 2017.
- [50] Z.-L. Tang, S. S. Ge, K. P. Tee, and W. He, "Robust adaptive neural tracking control for a class of perturbed uncertain nonlinear systems with state constraints," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 12, pp. 1618–1629, 2016.
- [51] Y.-J. Liu and S. Tong, "Barrier Lyapunov functions for Nussbaum gain adaptive control of full state constrained nonlinear systems," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 76, pp. 143–152, 2017.

- [52] Y.-J. Liu and S. Tong, "Barrier Lyapunov functions-based adaptive control for a class of nonlinear pure-feedback systems with full state constraints," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 64, pp. 70–75, 2016.
- [53] Y. J. Liu, S. Tong, C. L. P. Chen, and D. J. Li, "Adaptive NN control using integral barrier lyapunov functionals for uncertain nonlinear block-triangular constraint systems," *IEEE Transactions on Cybernetics*.
- [54] Y. J. Liu, S. M. Lu, D. J. Li, and S. C. Tong, "Adaptive controller design-based ABLF for a class of nonlinear time-varying state constraint systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 7, pp. 1546–1553, 2017.
- [55] Y.-J. Liu and S. Tong, "Barrier Lyapunov functions-based adaptive control for a class of nonlinear pure-feedback systems with full state constraints," *Automatica*, vol. 64, pp. 70–75, 2016.
- [56] L. Ding, S. Li, Y. J. Liu, H. B. Gao, C. Chen, and Z. Q. Deng, "Adaptive neural network-based tracking control for full-state constrained wheeled mobile robotic system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2410–2419, 2017.

Research Article

Forecasting the Acquisition of University Spin-Outs: An RBF Neural Network Approach

Weiwei Liu,^{1,2} Zhile Yang,³ and Kexin Bi^{1,4}

¹School of Economics and Management, Harbin Engineering University, Harbin 150001, China

²Management School, Queen's University Belfast, Belfast BT9 5EE, UK

³Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China

⁴School of Management, Harbin University of Science and Technology, Harbin 150080, China

Correspondence should be addressed to Zhile Yang; zyang07@qub.ac.uk

Received 14 July 2017; Accepted 14 September 2017; Published 17 October 2017

Academic Editor: Jing Na

Copyright © 2017 Weiwei Liu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

University spin-outs (USOs), creating businesses from university intellectual property, are a relatively common phenomena. As a knowledge transfer channel, the spin-out business model is attracting extensive attention. In this paper, the impacts of six equities on the acquisition of USOs, including founders, university, banks, business angels, venture capitals, and other equity, are comprehensively analyzed based on theoretical and empirical studies. Firstly, the average distribution of spin-out equity at formation is calculated based on the sample data of 350 UK USOs. According to this distribution, a radial basis function (RBF) neural network (NN) model is employed to forecast the effects of each equity on the acquisition. To improve the classification accuracy, the novel set-membership method is adopted in the training process of the RBF NN. Furthermore, a simulation test is carried out to measure the effects of six equities on the acquisition of USOs. The simulation results show that the increase of university's equity has a negative effect on the acquisition of USOs, whereas the increase of remaining five equities has positive effects. Finally, three suggestions are provided to promote the development and growth of USOs.

1. Introduction

Universities have the functions of talents training and scientific research. Moreover, policymakers, on the other hand, have increasing concerns on what the role universities can play in economic development. Universities are requested to make contributions to their region's economic development by promoting effective university-industry relationships to exploit and commercialize research discoveries. Exploiting scientific and technological developments in universities is a main theme in economic and industrial policy [1]. However, universities may not be able to capture the full value of their technology due to the limitation of the licensing arrangement. Therefore, spinning out a company becomes a more direct involvement in the commercialization of new technology [2]. Accordingly, there is a shift in emphasis from exploiting university intellectual property by licensing to a focus on

spin-out activities [3], and to stimulate spin-outs has become a significant issue for both universities and governments [4].

However, USOs are familiar with other small high-tech firms and of poor ability to debt financing and also depend highly on financing equity for their growth. Therefore, as a financing channel, equity distribution of USOs is attracting extensive attention. *Radial basis function* (RBF) *neural network* (NN) method is a novel and effective feedforward neural network with best approximation and global optimal performance. The training method is fast and easy and has been successfully applied in many fields with its unique information processing capability, especially in the financial, economic, and management fields [5–8]. This study incorporates RBF NN to investigate the impacts of equity distribution on the acquisition of USOs, aiming to deliver quantitative references for policymakers, founders, and financing providers to promote the development and growth of USOs.

Three major contributions of this study are concluded as follows:

- (1) It is the first attempt to adopt RBF NN method for forecasting the acquisition of USOs, which can resolve the problems that are intractable for traditional statistical method.
- (2) To improve the classification accuracy, a novel set-membership method is adopted in the training process of the RBF NN.
- (3) The simulation results confirm the effects of six equities on the acquisition of USOs and therefore provide quantitatively management decision references for policymakers, founders, universities, financial providers, and so forth.

The remainder of the paper is organized as follows. Section 2 provides an overview of previous studies on USOs and RBF NN, followed by Section 3 in which data and methodology are demonstrated. Section 4 presents the numerical results and discussions. Section 5 concludes the paper with the major findings from the methods and their policy implications.

2. Literature Review

2.1. University Spin-Outs (USOs). Lockett and Wright (2005) defined USOs as new ventures that are dependent upon licensing or assignment of an institution's intellectual property for initiation [9]. Moreover, USOs can also be defined as firms established by university academics aiming at commercializing ideas based on scientific discoveries and inventions [10, 11]. For many years, various definitions of university spin-outs have emerged, and scholars generally agree that the spin-outs derive from technologies developed within a university and the individuals who pursue their commercialization including scientific researchers, students, and graduates (Benner and Tushman, 2003 [12]; Steffensen et al., 2000 [13]; Klofsten and Jones-Evans, 2000 [14]). As a knowledge transfer channel, spin-out strategy is a key issue. Siegel et al. (2003) [3] pointed to the shift in emphasis from exploiting university intellectual property through licensing to a focus on spin-out activities.

The literatures in terms of spin-outs generally identify two categories of finance funding and support in the process of spin-out creation (De Coster and Butler, 2005 [15]; Landry et al., 2007 [16]): business angels and venture capitalists. Several evidences show a limited role for bank finance in spin-out creation in both UK and US studies (Roberts, 1991 [17]; EC, 2000a, b [18, 19]). Lockett and Wright (2005) stated that, in permitted conditions, the university may own equity in the spin-out, and some universities state that they encourage academic entrepreneurship and demand a share of the spin-out equity, and there is a positive relation between new venturing and the involvement of academic founders in the spin-out equity (Muscio et al., 2016) [20].

2.2. Radial Basis Function (RBF) Neural Network (NN). Nowadays, RBF method has been a well-adopted tool in the

stock market forecast and the prediction of nonlinear system [21–23]. An increasingly popular and promising approach to solve option pricing models is the use of numerical methods based on RBF [24]. Chan and Hubbert (2014) [25] demonstrated how European and American option prices can be computed under the jump-diffusion model using the RBF interpolation scheme. An implementation of RBF method for solving Black-Scholes-type partial differential equations (PDEs) is proposed to price the swaptions in the absence of credit risk [26]. Erdal and Ekinici make a comparison of various artificial intelligence methods in the prediction of bank failures, including support vector machines (SVMs), RBF NN, and multilayer perceptrons (MLPs), in addition to subjecting the explanatory variables to principal component analysis (PCA) [27]. RBF is also used for the valuation, optimization, market, margin, and credit risk management of gas-fired power plants and associated tolling contracts [28]. A four-phase dynamic feedback RBF model is established for supply partner selection in agile supply chains (ASCs) [29]. Kapetanios and Blake (2010) [30] propose new tests for the martingale difference restriction based on RBF NN. In summary, it can be seen that RBF neural network is widely used and developed in the fields of economy, management, finance, social sciences, and so forth.

Though the RBF has seen numerous applications in economic and management modeling areas, according to the authors' knowledge, it has not been found to be of use and of significant potentials in USOs relevant researches.

3. Data and Methodology

3.1. Data. In an effort to enhance our understanding of this sector, a database of 1044 active USOs was compiled from individual university records and Internet searches and matched to a published list of UK university spin-outs [31]. Telephone interviews were conducted with UK USOs and a final sample of 350 was achieved. And finally we obtained several parts of the data, including finance, incubation, support, and intellectual property. According to the characteristics of the collected data, traditional statistical analysis method could not solve the problem well; therefore, this study attempts to use RBF neural network to test the influences of equity distribution on the acquisition of USOs.

In our survey of UK USOs, the distribution of equity was measured in 2014 when the survey was not only conducted, but also traced back when the business was formed. At the point when USOs are formed, on average the founders own 56.51 (SD = 30.92) percent of equity, and 24.73 (SD = 23.36) percent belongs to the host university. Financial providers own a relatively small share of the overall equity at this stage of development, with venture capitals accounting for 7.71 (SD = 16.90) percent followed by business angels at 2.24 (SD = 8.37) percent and, to a much less extent, banks at 0.45 (SD = 4.53) percent and others at 8.36 (SD = 21.11) percent, as shown in Figure 1.

3.2. RBF Neural Network. RBF NN is a three-layer forward network, including input-layer, hidden-layer and output-layer, as shown in Figure 2. The performance of RBF NN

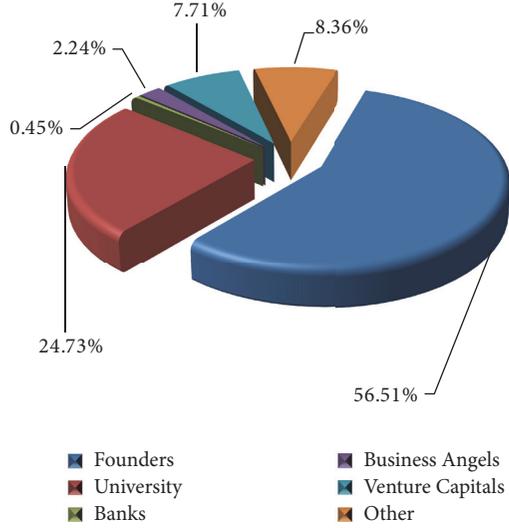


FIGURE 1: Average distribution of SPOs equity at formation (percentage).

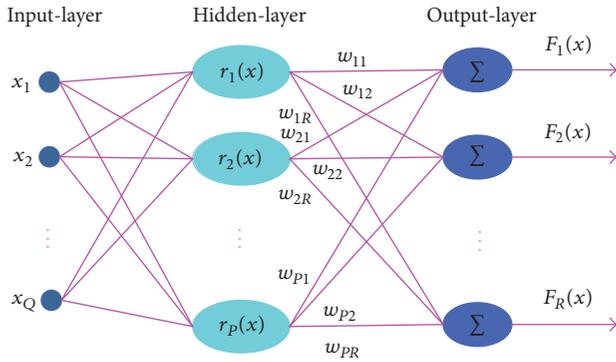


FIGURE 2: RBF NN model.

is determined by its topological parameters, such as the mean and variance of the RBF, the number of hidden nodes, and the weight between the hidden-layer and the output-layer. Therefore, the training process of RBF NN focuses on adjusting the aforementioned factors so as to obtain precise model between inputs and outputs. Specifically, the location of RBF in the input space is influenced by its mean value, while the active zone is affected by its variance. The RBF or the neuron would be activated and produce large output if the input signal is close to its center. Otherwise, the output of RBF will tend to zero. Due to the limited response region of each RBF, the linear projection from hidden-layer to output-layer is needed to make the whole NN achieve specific function, such as classification and approximation [32].

The number of hidden nodes is a crucial factor in RBF NN: too little hidden nodes limit the learning ability, while a large number of hidden nodes lead to increasing training time and overfitting. According to the number of hidden nodes, the RBF NN is classified to generalized network (GN) model and regularized network (RN) model. By setting a restriction with prior knowledge, a smooth mapping function is obtained. The GN is suitable for classification. The input

vectors in low-dimension space are transformed by the hidden-layer of NN, composed of RBF, achieving the projection to high-dimension space. Then, a linear-inseparable problem in low dimensional space becomes separable in high-dimension space.

Based on the study on the single-layer perceptron, a hyperplane described by linear equations exists if the modes in training sample space are linear-separable. This hyperplane would not exist when the training samples are nonlinear-separable. However, according to Cover theorem [33], the nonlinear-separable problem could be solved by nonlinear transformation. This nonlinear transformation is conducted by the hidden-layer of RBF NN.

The most commonly used radial base function is Gaussian function. If the hidden nodes number is p , the activation function of the p th node is expressed in

$$r_p(x) = \exp\left(-\frac{\|x - \mu_p\|_2^2}{2\sigma_p^2}\right), \quad (1)$$

where $\|\cdot\|_2$ denotes the norm of the vector, μ_p ($p = 1, 2, \dots, P$) denotes the center of the radial base function, and σ_p denotes the variance of the Gaussian function. Suppose the numbers of neuron in input-layer and output-layer are Q and R , respectively. Correspondingly, the input vector is $x = [x_1, x_2, \dots, x_Q]^T$ and the output vector is $y = [y_1, y_2, \dots, y_R]^T$. The activation functions of input-layer and output-layer are set to be identity function, and all the neuron thresholds are set to be zero. The weight between the hidden-layer and the output-layer could be expressed as

$$W = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1R} \\ w_{21} & w_{22} & \cdots & w_{2R} \\ \vdots & \vdots & \ddots & \vdots \\ w_{p1} & w_{p2} & \cdots & w_{pR} \end{bmatrix}. \quad (2)$$

Then the output of the r th neuron in the output-layer is

$$y_r = \sum_{p=1}^R w_{pr} r_p(x) = \sum_{p=1}^R w_{pr} \exp\left(-\frac{\|x - \mu_p\|_2^2}{2\sigma_p^2}\right). \quad (3)$$

Once the topology of the RBF NN is determined, the accuracy of classification is depending on the coefficients matrix W , which could be optimized by the training process [34, 35]

3.3. Set-Membership Training Method. Consider the following linear regression:

$$y_k = G(x_k)\theta + \varsigma_k, \quad k = 1, 2, \dots, n, \quad (4)$$

where $y_k \in \mathfrak{R}^n$ is the output, $x_k \in \mathfrak{R}^m$ is the input, ς_k is the noise which is bounded by $\|\varsigma_k\|_2 \leq r$, $\theta \in \mathfrak{R}^q$ is the parameter vector to be estimated, and $G(x_k) = [r_1(x_k), \dots, r_p(x_k)] \in \mathfrak{R}^{n \times q}$. For the simplicity of the presentation, we use G_k to denote $G(x_k)$ in the rest of the paper.

Let θ_k be the estimate of θ ; the output of RBF neural network is

$$\hat{y}_k = G_k \theta_{k-1}. \quad (5)$$

We define the training error as

$$e_k = y_k - \hat{y}_k. \quad (6)$$

θ_k can be computed such that the loss function

$$J_k = \sum_{i=1}^k \beta_i \frac{1}{r^2} \|e_i\|_2^2 \quad (7)$$

is minimized, where β_i is the weighted factor to be determined.

The corresponding estimate is given by

$$\theta_k = P_k \Phi_k \quad (8)$$

with

$$P_k^{-1} = \sum_{i=1}^k \beta_i \frac{1}{r^2} G_i^T G_i \quad (9)$$

$$\Phi_k = \sum_{i=1}^k \beta_i \frac{1}{r^2} G_i^T y_i.$$

We rewrite the above algorithm in the recursive form:

$$\theta_k = \theta_{k-1} + \beta_k \frac{1}{r^2} P_k G_k^T e_k, \quad (10)$$

$$P_k^{-1} = P_{k-1}^{-1} + \beta_k \frac{1}{r^2} G_k^T G_k \quad (11)$$

which is also equivalent to the algorithm in the following Kalman filter form:

$$\theta_k = \theta_{k-1} + \Gamma_k e_k \quad (12)$$

$$\Gamma_k = \beta_k P_{k-1} G_k^T (\beta_k G_k P_{k-1} G_k^T + r^2 I)^{-1} \quad (13)$$

$$P_k = (I - \Gamma_k G_k) P_{k-1}. \quad (14)$$

Until now, the design of the weighted factor β_k is undetermined. It should be pointed out that the purpose of introducing the weighted factor β_k is to measure the credibility of data y_k . If y_k is more credible than y_{k-1} , the weighted factor β_k should be larger than β_{k-1} . For instance, $\beta_k = 0$ indicates that y_k is not credible and should be neglected. For the design of the weighted factor β_k , the set-membership approach is employed. By reviewing the linear regression (4) and the boundedness property of the noise, θ is obviously in the set of the following form:

$$\left\{ \theta \in \mathfrak{R}^q : \frac{1}{r^2} (y_i - G_i \theta)^T (y_i - G_i \theta) \leq 1 \right\}. \quad (15)$$

For such k sets of data, θ is then in the intersection of k sets like (15), which can be expressed by

$$\Omega(k) = \bigcap_{i=1}^k \left\{ \theta \in \mathfrak{R}^q : \frac{1}{r^2} (y_i - G_i \theta)^T (y_i - G_i \theta) \leq 1 \right\}. \quad (16)$$

Note that it is difficult to describe $\Omega(k)$ mathematically. However, it is convenient to design a set $\Theta(k)$ that contains $\Omega(k)$,

$$\Theta(k) = \left\{ \theta \in \mathfrak{R}^q : \sum_{i=1}^k \beta_i \frac{1}{r^2} (y_i - G_i \theta)^T (y_i - G_i \theta) \leq \sum_{i=1}^k \beta_i \right\}. \quad (17)$$

Lemma 1. Given (12)–(14), the set $\Theta(k)$ given by (17) can be converted into the following ellipsoid expression:

$$\Theta(k) = \left\{ \theta \in \mathfrak{R}^q : (\theta - \theta_k)^T P_k^{-1} (\theta - \theta_k) \leq \sigma_k^2 \right\}. \quad (18)$$

Moreover, the recursive expression of σ_k^2 can be shown as below:

$$\sigma_k^2 = \sigma_{k-1}^2 + \beta_k \left[1 - e_k^T (\beta_k G_k P_{k-1} G_k^T + r^2 I)^{-1} e_k \right]. \quad (19)$$

Proof. Expanding (17) yields to

$$\sum_{i=1}^k \beta_i \frac{1}{r^2} \theta^T G_i^T G_i \theta - \sum_{i=1}^k \beta_i \frac{1}{r^2} y_i^T G_i \theta - \sum_{i=1}^k \beta_i \frac{1}{r^2} \theta^T G_i^T y_i \leq \sum_{i=1}^k \beta_i \left(1 - \frac{1}{r^2} y_i^T y_i \right). \quad (20)$$

Substituting (9) into (20) leads to

$$\theta^T P_k^{-1} \theta - \Phi_k^T \theta - \theta^T \Phi_k \leq \sum_{i=1}^k \beta_i \left(1 - \frac{1}{r^2} y_i^T y_i \right). \quad (21)$$

Adding $\theta_k^T P_k^{-1} \theta_k$ to both sides of (21) leads to

$$\begin{aligned} & \theta^T P_k^{-1} \theta - \Phi_k^T \theta - \theta^T \Phi_k + \theta_k^T P_k^{-1} \theta_k \\ & \leq \sum_{i=1}^k \beta_i \left(1 - \frac{1}{r^2} y_i^T y_i \right) + \theta_k^T P_k^{-1} \theta_k. \end{aligned} \quad (22)$$

Consider $\Phi_k = P_k^{-1} \theta_k$. The left side of (22) equals

$$\begin{aligned} & \theta^T P_k^{-1} \theta - \Phi_k^T \theta - \theta^T \Phi_k + \theta_k^T P_k^{-1} \theta_k \\ & = (\theta - \theta_k)^T P_k^{-1} (\theta - \theta_k). \end{aligned} \quad (23)$$

Define the right side of (22) as the radius of the ellipsoid

$$\sigma_k^2 = \sum_{i=1}^k \beta_i \left(1 - \frac{1}{r^2} y_i^T y_i \right) + \theta_k^T P_k^{-1} \theta_k. \quad (24)$$

Thus, we can express $\Theta(k)$ by (15) in the ellipsoid form.

Moreover, from (24) the recursive form of σ_k^2 equals

$$\begin{aligned} \sigma_k^2 &= \sigma_{k-1}^2 + \beta_k \left(1 - \frac{1}{r^2} y_k^T y_k \right) + \theta_k^T P_k^{-1} \theta_k \\ &\quad - \theta_{k-1}^T P_{k-1}^{-1} \theta_{k-1}. \end{aligned} \quad (25)$$

Consider $\Phi_k = P_k^{-1}\theta_k$ and (10). The sum of the last two terms of (25) equals

$$\begin{aligned} \theta_k^T P_k^{-1} \theta_k - \theta_{k-1}^T P_{k-1}^{-1} \theta_{k-1} &= \theta_k^T \Phi_k - \theta_{k-1}^T \Phi_{k-1} \\ &= \theta_{k-1}^T \Phi_k + \beta_k \frac{1}{r^2} e_k^T G_k P_k \Phi_k - \theta_{k-1}^T \Phi_{k-1}. \end{aligned} \quad (26)$$

Since $\Phi_k = \Phi_{k-1} + \beta_k \frac{1}{r^2} G_k^T y_k$, (26) further gives

$$\begin{aligned} \theta_k^T P_k^{-1} \theta_k - \theta_{k-1}^T P_{k-1}^{-1} \theta_{k-1} &= \beta_k \frac{1}{r^2} \theta_{k-1}^T G_k^T y_k + \beta_k \frac{1}{r^2} e_k^T G_k \theta_{k-1} \\ &\quad + \beta_k^2 \frac{1}{r^4} e_k^T G_k P_k G_k^T e_k. \end{aligned} \quad (27)$$

Substituting (27) into (25) results in

$$\begin{aligned} \sigma_k^2 &= \sigma_{k-1}^2 + \beta_k - \beta_k \frac{1}{r^2} e_k^T e_k + \beta_k^2 \frac{1}{r^4} e_k^T G_k P_k G_k^T e_k \\ &= \sigma_{k-1}^2 + \beta_k - \beta_k \frac{1}{r^2} e_k^T (I - G_k \Gamma_k) e_k \\ &= \sigma_{k-1}^2 + \beta_k \left[1 - e_k^T (\beta_{k-1} G_{k-1} P_{k-1} G_{k-1}^T + r^2 I)^{-1} e_k \right]. \end{aligned} \quad (28)$$

That is the end of the proof. \square

On the basis of Lemma 1, we further transform (19) into that

$$\begin{aligned} \sigma_k^2 &\leq \sigma_{k-1}^2 \\ &\quad + \beta_k \left[1 - \lambda_{\min} \left\{ (\beta_k G_k P_{k-1} G_k^T + r^2 I)^{-1} \right\} e_k^T e_k \right] \\ &= \sigma_{k-1}^2 + \beta_k \left[1 - \frac{e_k^T e_k}{\lambda_{\max} \{ \beta_k G_k P_{k-1} G_k^T + r^2 I \}} \right] \\ &= \sigma_{k-1}^2 + \beta_k \left[1 - \frac{e_k^T e_k}{(\beta_k \|G_k P_{k-1} G_k^T\|_2 + r^2)} \right]. \end{aligned} \quad (29)$$

Finally, we design β_k as follows:

$$\beta_k = \begin{cases} \frac{(\|e_k\|_2^2 - r^2)}{\|G_k P_{k-1} G_k^T\|_2} & \|e_k\|_2 \geq r \\ 0 & \text{otherwise.} \end{cases} \quad (30)$$

Based on the above development, the main properties of the proposed training algorithm given by (12)–(14) are summarized in the following theorem.

Theorem 2. Consider the training algorithm given by (12)–(14), if we design the weighted factor β_k according to (30), then the following statements are verified:

- (1) If $\theta \in \Theta(0)$, then $\theta \in \Theta(k)$, $\forall k$.
- (2) $\|\theta - \theta_k\|_2$ is bounded and nonincreasing.
- (3) $\lim_{k \rightarrow \infty} \|e_k\|_2 \leq r$.

Proof. (1) Define $\tilde{\theta}_k = \theta - \theta_k$ and choose the following Lyapunov function:

$$V_k = \tilde{\theta}_k^T P_k^{-1} \tilde{\theta}_k. \quad (31)$$

Substitution of (10) into the Lyapunov function leads to

$$\begin{aligned} V_k &= \left(\tilde{\theta}_{k-1} - \beta_k \frac{1}{r^2} P_k G_k^T e_k \right)^T P_k^{-1} \left(\tilde{\theta}_{k-1} - \beta_k \frac{1}{r^2} P_k G_k^T e_k \right) \\ &= \tilde{\theta}_{k-1}^T P_k^{-1} \tilde{\theta}_{k-1} - \beta_k \frac{1}{r^2} e_k^T G_k \tilde{\theta}_{k-1} - \beta_k \frac{1}{r^2} \tilde{\theta}_{k-1}^T G_k^T e_k \\ &\quad + \beta_k^2 \frac{1}{r^4} e_k^T G_k P_k G_k^T e_k. \end{aligned} \quad (32)$$

Recalling (11), (32) is expanded to

$$\begin{aligned} V_k &= V_{k-1} + \beta_k \frac{1}{r^2} \tilde{\theta}_{k-1}^T G_k^T G_k \tilde{\theta}_{k-1} - \beta_k \frac{1}{r^2} e_k^T G_k \tilde{\theta}_{k-1} \\ &\quad - \beta_k \frac{1}{r^2} \tilde{\theta}_{k-1}^T G_k^T e_k + \beta_k^2 \frac{1}{r^4} e_k^T G_k P_k G_k^T e_k. \end{aligned} \quad (33)$$

Since $\varsigma_k = e_k - G_k \tilde{\theta}_{k-1}$, (33) is equivalent to

$$\begin{aligned} V &= V_{k-1} + \beta_k \frac{1}{r^2} \varsigma_k^T \varsigma_k - \beta_k \frac{1}{r^2} e_k^T e_k \\ &\quad + \beta_k^2 \frac{1}{r^4} e_k^T G_k P_k G_k^T e_k \\ &= V_{k-1} + \beta_k \frac{1}{r^2} \varsigma_k^T \varsigma_k \\ &\quad - \beta_k \frac{1}{r^2} e_k^T \left(I - \beta_k \frac{1}{r^2} G_k P_k G_k^T \right) e_k \\ &= V_{k-1} + \beta_k \frac{1}{r^2} \varsigma_k^T \varsigma_k - \beta_k \frac{1}{r^2} e_k^T (I - G_k \Gamma_k) e_k. \end{aligned} \quad (34)$$

Substitution of the learning gain matrix presented by (13) yields to

$$V_k \leq V_{k-1} + \beta_k \left[1 - e_k^T (\beta_k G_k P_{k-1} G_k^T + r^2 I)^{-1} e_k \right]. \quad (35)$$

In observing (29), (35) equals

$$V_k \leq V_{k-1} + \sigma_k^2 - \sigma_{k-1}^2 \quad (36)$$

which can be rewritten as

$$V_k - \sigma_k^2 \leq V_{k-1} - \sigma_{k-1}^2. \quad (37)$$

Then it follows that $V_{k-1} \leq \sigma_{k-1}^2$ deduces $V_k \leq \sigma_k^2$; namely, $\theta \in \Theta(k-1)$ deduces $\theta \in \Theta(k)$.

(2) From the proof of the statement (1), it follows that

$$\begin{aligned} V_k &\leq V_{k-1} + \beta_k \left[1 - e_k^T (\beta_k G_k P_{k-1} G_k^T + r^2 I)^{-1} e_k \right] \\ &\leq V_{k-1} + \beta_k \left[1 - \frac{e_k^T e_k}{(\beta_k \|G_k P_{k-1} G_k^T\|_2 + r^2)} \right]. \end{aligned} \quad (38)$$

Considering the design of β_k given by (30), (38) further gives

$$V_k \leq V_{k-1} \quad (39)$$

which is equivalent to

$$\tilde{\theta}_k^T P_k^{-1} \tilde{\theta}_k \leq \tilde{\theta}_{k-1}^T P_{k-1}^{-1} \tilde{\theta}_{k-1}. \quad (40)$$

Noting $P_k^{-1} \geq P_{k-1}^{-1}$, the following inequality holds

$$\tilde{\theta}_k^T P_{k-1}^{-1} \tilde{\theta}_k \leq \tilde{\theta}_{k-1}^T P_{k-1}^{-1} \tilde{\theta}_{k-1}. \quad (41)$$

Thus $\tilde{\theta}_k^T \tilde{\theta}_k \leq \tilde{\theta}_{k-1}^T \tilde{\theta}_{k-1}$ is proved.

(3) The statement (3) will be verified for the following different cases.

Firstly, when $\lim_{k \rightarrow \infty} \sum_{i=1}^k \beta_i = \infty$, it results in $\lim_{k \rightarrow \infty} P_k^{-1} = \infty$ according to (6). Then we know that $\lim_{k \rightarrow \infty} \tilde{\theta}_k^T \tilde{\theta}_k = 0$, due to $V_k \leq V_{k-1} \leq V_0 < \infty$ which can be observed from the proof of statement (2). In this case, the statement (3) is straightforward.

Secondly, when $\lim_{k \rightarrow \infty} \sum_{i=1}^k \beta_i < \infty$, it leads to $\lim_{k \rightarrow \infty} \beta_k = 0$. Thus, $\lim_{k \rightarrow \infty} \|e_k\|_2 \leq r$ can be derived from (30).

That is the end of the entire proof. \square

Remark 3. The theoretical proof of this paper shows that, as the input data increases, the estimation values of the parameters will gradually approximate the true values of the parameters. This proof ensures the convergence of the estimation error.

4. Results and Discussions

We use the typical three-layer RBF neural network for the classification of 350 sets of data. In the RBF NN model, the input vector is six-dimensional and the output data is one-dimensional vector. We assume that there are 19 neural hidden nodes, and the width value of the Gaussian function for each neural net is initially chosen as 10. The center point of the Gaussian function for the i th neural net is chosen as $i \times [5, 5, 5, 5, 5, 5]^T$, $i = 1, 2, \dots, 19$. The weight value vector is trained by the proposed set-membership approach and the classification result can be shown in Figures 3 and 4.

The vertical coordinates of Figures 3 or 4 denote the output values of the neural networks, which represent the possibilities of the companies being acquired. And the green diamonds in Figures 3 and 4 represent the acquired companies, and the red circles in Figure 3 represent the companies that have not been acquired.

The closer the output value is to 1, the more likely the company is to be acquired. The closer the output value is to 0, the less likely the company is to be acquired. The output value of neural network is continuous, but the acquisition is a Boolean variable. In order to achieve the purpose of the classification, we choose 0.5 as the dividing line (the blue line in Figures 3 and 4). The output value greater than 0.5 is on behalf of the company being acquired; otherwise the company will not be acquired. By comparing the neural

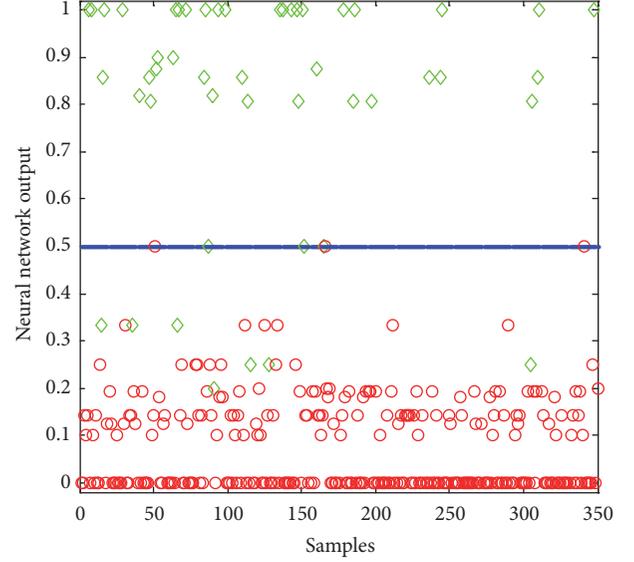


FIGURE 3: The RBF NN model training result (accuracy: 97.4%).

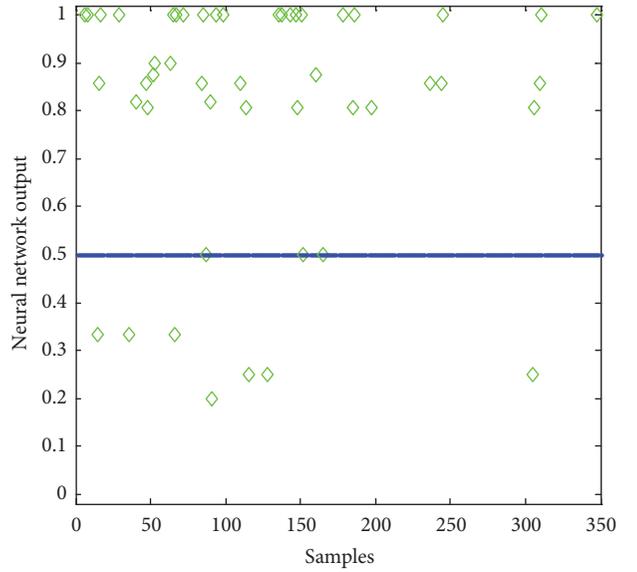


FIGURE 4: The RBF NN model classification result (accuracy: 81.6%).

network training results with the real acquisition results, it could be observed from Figures 3 and 4 that the model training accuracy has achieved 97.4%, whereas the validation classification result is 81.6%. It should be noted that the less accuracy of the validation is mainly due to the fact that the validation data resource is only from the acquired spin-outs which may not have fully representing behaviors for all spin-outs. The weight value vector has been obtained by means of the set-membership approach proposed in Section 3.3. With the trained weight value, the average value of the NN output is computed and equals 0.14. Then, we may wonder how the following six factors (equity distributions) contribute to the acquisition of USOs. Thus, we modify the original data sets by increasing proportion of one of the factors and decreasing the ratio of other factors correspondingly. Then, the average value

TABLE 1: Contribution results of different factors.

Factor	Change of average value	Positive or negative	Strong, medium, or weak
Founders	+0.0448	Positive	Weak
University	-0.0612	Negative	Medium
Banks	+0.1228	Positive	Strong
Business angels	+0.0406	Positive	Weak
Venture capitals	+0.0813	Positive	Medium
Other	+0.1193	Positive	Strong

is computed by testing the modified data sets with the same weight value. If the average value increases, we believe that this factor contributes to the acquisition; otherwise this factor will not contribute to the acquisition or even discourage the acquisition. Meanwhile, the average value increases more meaning that the corresponding factor contributes to the acquisition more. The results are as shown in Table 1.

From the simulation results, it can be found that the increase of founders' equity has a positive effect on the acquisition of USOs, though the effect is weak; the increase of university's equity has a negative effect on the acquisition of USOs, and the effect is medium; the increase of banks' equity has a strong positive effect on the acquisition of USOs by the value as +0.1228; the increase of business angels' equity has a relatively weak positive effect on the acquisition of USOs, with the change value as +0.0406; the increase of venture capitals' equity has a positive effect on the acquisition of USOs, and the effect is medium; the increase of other equity has a positive effect on the acquisition of USOs, and the effect is strong.

It is demonstrated that the increases of founders, financial providers (including banks, business angels, and venture capitals), and other equity are conducive to the acquisition of USOs, and banks' equity is the most influential factor for the acquisition. The main reason for the results is possibly due to the fact that founders and financial are more concerned about the benefits and profits of USOs; therefore, the increase of their equities is inclined to the acquisition. The reason why other equity has a positively strong effect on the acquisition might likely be that this is captured by other businesses, who may be forming strategic joint ventures or alliances with the USOs. The effect of business angels' equity is much less, and probably the reason for this is that business angels come from diverse backgrounds ranging from former entrepreneurs to finance specialists, all of whom are inclined to target the less risky proposals compared to those favored by venture capitalists. On the other hand, the increase of university equity or other equity is not conducive to be acquired for USOs, and it could be explained that universities are more concerned about the emergence and transfer of knowledge and technology compared to the economic benefits.

5. Conclusions and Suggestions

This paper adopts RBF NN model to investigate the impacts of six equities (including founders, university, banks, business

angels, venture capitals, and others) on the acquisition of USOs. Based on the sample data of 350 UK USOs, the average distribution of spin-out equity at formation is first calculated. According to this distribution, an RBF NN model is employed to forecast the effects of each equity change on the acquisition. To improve the classification accuracy, the novel set-membership method is adopted in the training process of the RBF NN. Furthermore, a simulation test is carried out to test the effects of six equities on the acquisition of USOs. The simulation results show that the increase of university's equity has a negative effect on the acquisition of USOs; however, the increase of remaining five equities has positive impact. The results demonstrate that the increases of founders, financial providers (including banks, business angels, and venture capitals), and other equity are conducive to the acquisition of USOs.

According to the key findings of this paper, some policy implications are proposed as references for decision-making by policymakers, founders, universities, financing providers, and so on.

- (1) If USOs would like to be acquired in the future, a reasonable equity allocation should be ensured by the time of USOs formation when creating businesses from university.
- (2) If the equity distribution was determined when the business was formed, and the USO has not been acquired all the time, they should adjust equity distribution and increase the percentage of founders' equity, financial providers' equity, or other equity appropriately.
- (3) In order to be acquired, USOs should encourage and explore multiple investments and financing channels.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research is financially supported by the National Natural Science Foundation of China under Grants 71302028, 71472057, and 71602041, Heilongjiang Province Natural Science Fund (LC2017031), and Fundamental Scientific Research Funds for the Central Universities (HEUCFI70903, HEUCFW170908, and HEUCFW170904).

References

- [1] R. T. Harrison and C. Leitch, "Voodoo institution or entrepreneurial university? spin-off companies, the entrepreneurial system and regional development in the UK," *Regional Studies*, vol. 44, no. 9, pp. 1241–1262, 2010.
- [2] S. J. Franklin, M. Wright, and A. Lockett, "Academic and surrogate entrepreneurs in university spin-out companies," *The Journal of Technology Transfer*, vol. 26, no. 1-2, pp. 127–141, 2001.
- [3] D. S. Siegel, D. Waldman, and A. Link, "Assessing the impact of organizational practices on the relative productivity of

- university technology transfer offices: an exploratory study," *Research Policy*, vol. 32, no. 1, pp. 27–48, 2003.
- [4] S. Hess and R. Y. Siegwart, "University technology incubator: technology transfer of early stage technologies in cross-border collaboration with industry," *Business and Management Research*, vol. 2, no. 2, pp. 22–36, 2013.
 - [5] C. L. Lin, J. F. Wang, C. Y. Chen, C. W. Chen, and C. W. Yen, "Improving the generalization performance of RBF neural networks using a linear regression technique," *Expert Systems with Applications*, vol. 36, no. 10, pp. 12049–12053, 2009.
 - [6] H. K. K. Tung and M. C. S. Wong, "On the formulation of credit barrier model using radial basis functions," *Journal of the Operational Research Society*, vol. 65, no. 9, pp. 1437–1452, 2014.
 - [7] A. Guarin, X. Liu, and W. L. Ng, "Enhancing credit default swap valuation with meshfree methods," *European Journal of Operational Research*, vol. 214, no. 3, pp. 805–813, 2011.
 - [8] P. A. Gutiérrez, M. J. Segovia-Vargas, S. Salcedo-Sanz et al., "Hybridizing logistic regression with product unit and RBF networks for accurate detection and prediction of banking crises," *Omega*, vol. 38, no. 5, pp. 333–344, 2010.
 - [9] A. Lockett and M. Wright, "Resources, capabilities, risk capital and the creation of university spin-out companies," *Research Policy*, vol. 34, no. 7, pp. 1043–1057, 2005.
 - [10] P. Mustar, M. Renault, M. G. Colombo et al., "Conceptualising the heterogeneity of research-based spin-offs: a multi-dimensional taxonomy," *Research Policy*, vol. 35, no. 2, pp. 289–308, 2006.
 - [11] A. Walter, M. Auer, and T. Ritter, "The impact of network capabilities and entrepreneurial orientation on university spin-off performance," *Journal of Business Venturing*, vol. 21, no. 4, pp. 541–567, 2006.
 - [12] M. J. Benner and M. L. Tushman, "Exploitation, exploration, and process management: the productivity dilemma revisited," *Academy of Management Review (AMR)*, vol. 28, no. 2, pp. 238–256, 2003.
 - [13] M. Steffensen, E. M. Rogers, and K. Speakman, "Spin-offs from research centers at a research university," *Journal of Business Venturing*, vol. 15, no. 1, pp. 93–111, 2000.
 - [14] M. Klofsten and D. Jones-Evans, "Comparing academic entrepreneurship in europe—The case of sweden and ireland," *Small Business Economics*, vol. 14, no. 4, pp. 299–309, 2000.
 - [15] R. De Coster and C. Butler, "Assessment of proposals for new technology ventures in the UK: characteristics of university spin-off companies," *Technovation*, vol. 25, no. 5, pp. 535–543, 2005.
 - [16] R. Landry, N. Amara, and M. Saïlhi, "Patenting and spin-off creation by Canadian researchers in engineering and life sciences," *The Journal of Technology Transfer*, vol. 32, no. 3, pp. 217–249, 2007.
 - [17] E. B. Roberts, *Entrepreneurs in High Technology*, Oxford University Press, New York, NY, USA, 1991.
 - [18] European Commission, Progress Report on the Risk Capital Action Plan, Brussels and Luxembourg, 2000a.
 - [19] European Commission, Innovation in a Knowledge Driven Economy, Brussels and Luxembourg, 2000b.
 - [20] A. Muscio, D. Quaglione, and L. Ramaciotti, "The effects of university rules on spinoff creation: the case of academia in Italy," *Research Policy*, vol. 45, no. 7, pp. 1386–1396, 2016.
 - [21] J. De Jesus Rubio, P. Angelov, and J. Pacheco, "Uniformly stable backpropagation algorithm to train a feedforward neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 22, no. 3, pp. 356–366, 2011.
 - [22] O. Abedinia and N. Amjady, "Net demand prediction for power systems by a new neural network-based forecasting engine," *Complexity*, vol. 21, no. S2, pp. 296–308, 2016.
 - [23] M.-C. Pai, "RBF-based discrete sliding mode control for robust tracking of uncertain time-delay systems with input nonlinearity," *Complexity*, vol. 21, no. 6, pp. 194–201, 2016.
 - [24] L. V. Ballestra and G. Pacelli, "Pricing European and American options with two stochastic factors: a highly efficient radial basis function approach," *Journal of Economic Dynamics and Control*, vol. 37, no. 6, pp. 1142–1167, 2013.
 - [25] R. T. L. Chan and S. Hubbert, "Options pricing under the one-dimensional jump-diffusion model using the radial basis function interpolation scheme," *Review of Derivatives Research*, vol. 17, no. 2, pp. 161–189, 2014.
 - [26] M. Thompson, "Counterparty credit risk pricing and measurement of swaption portfolios," *The Journal of Computational Finance*, vol. 18, no. 1, pp. 31–64, 2014.
 - [27] H. I. Erdal and A. Ekinçi, "A comparison of various artificial intelligence methods in the prediction of bank failures," *Computational Economics*, vol. 42, no. 2, pp. 199–215, 2013.
 - [28] M. Thompson, "Optimal economic dispatch and risk management of thermal power plants in deregulated markets," *Operations Research*, vol. 61, no. 4, pp. 791–809, 2013.
 - [29] C. Wu and D. Barnes, "A dynamic feedback model for partner selection in agile supply chains," *International Journal of Operations and Production Management*, vol. 32, no. 1, pp. 79–103, 2012.
 - [30] G. Kapetanios and A. P. Blake, "Tests of the martingale difference hypothesis using boosting and RBF neural network approximations," *Econometric Theory*, vol. 26, no. 5, pp. 1363–1397, 2010.
 - [31] <http://www.spinoutsuk.co.uk/listings/company-listings/>.
 - [32] D. Chen and W. Han, "Prediction of multivariate chaotic time series via radial basis function neural network," *Complexity*, vol. 18, no. 4, pp. 55–66, 2013.
 - [33] N. Q. Huy, D. S. Kim, and N. V. Tuyen, "Existence theorems in vector optimization with generalized ORDer," *Journal of Optimization Theory and Applications*, vol. 174, no. 3, pp. 728–745, 2017.
 - [34] W. Yu and J. D. J. Rubio, "Recurrent neural networks training with stable bounding ellipsoid algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 20, no. 6, pp. 983–991, 2009.
 - [35] D. J. R. Jose, Y. Wen, and F. Andres, "Neural network training with optimal bounded ellipsoid algorithm," *Neural Computing and Applications*, vol. 18, pp. 623–631, 2009.

Research Article

A Novel SHLNN Based Robust Control and Tracking Method for Hypersonic Vehicle under Parameter Uncertainty

Chuanfeng Li,^{1,2} Hao Wu,³ Zhile Yang,⁴ Yongji Wang,⁵ and Zeyu Sun¹

¹School of Computer and Information Engineering, Luoyang Institute of Science and Technology, Luoyang 471023, China

²School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Belfast BT9 5AH, UK

³Beijing Aerospace Automation Control Institute, Beijing 100854, China

⁴Shenzhen Institute of Advanced Technology, Chinese Academy of Sciences, Shenzhen, Guangdong 518055, China

⁵School of Automation, Huazhong University of Science and Technology, Wuhan 430074, China

Correspondence should be addressed to Zhile Yang; zyang07@qub.ac.uk

Received 6 July 2017; Accepted 11 September 2017; Published 17 October 2017

Academic Editor: Guang Li

Copyright © 2017 Chuanfeng Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Hypersonic vehicle is a typical parameter uncertain system with significant characteristics of strong coupling, nonlinearity, and external disturbance. In this paper, a combined system modeling approach is proposed to approximate the actual vehicle system. The state feedback control strategy is adopted based on the robust guaranteed cost control (RGCC) theory, where the Lyapunov function is applied to get control law for nonlinear system and the problem is transformed into a feasible solution by linear matrix inequalities (LMI) method. In addition, a nonfragile guaranteed cost controller solved by LMI optimization approach is employed to the linear error system, where a single hidden layer neural network (SHLNN) is employed as an additive gain compensator to reduce excessive performance caused by perturbations and uncertainties. Simulation results show the stability and well tracking performance for the proposed strategy in controlling the vehicle system.

1. Introduction

The long-distance unpowered glide reentry vehicle is an important hypersonic vehicle which has been of significant aerodynamic configuration with high lift-to-drag ratio. It can reach the target after long-distance gliding and fulfill a throwing mission through reentering from orbit or suborbit. Given strong ability to fulfill high-speed remote precision attack and power projection, this vehicle is of great implication function for strategic planning. However, the vehicle is a complex nonlinear object, and how to design the control strategy to ensure the stability of vehicle system has become a crucial topic [1–6].

Conventional technologies are majorly based on performing time-domain simulation and relied heavily on the results of human experience. Since the birth of modern control theory in the 1950s, control theory develops rapidly and has been successfully adopted in the aerospace application in the 1960s [7, 8]. In the recent few decades, robust control has gained remarkable attentions due to the well

adaptation ability in dealing with objects in uncertain and noisy environment [9–13]. With the maturation of robust control theory, Kharitonov interval theory, H_∞ control theory, and structural singular value theory (μ theory) have been widely used in aircraft controller design and trajectory tracking. For instance, the refinement of the existing method by considering 16 segment plants instead of 16 Kharitonov plants provides an efficient tool for designing all robustly stabilizing PID controllers for an interval system [14]. An H_∞ method for designing reduced-order output-feedback controllers for linear time-invariant retarded systems was introduced to achieve a minimum bound on the H -infinity performance level [15]. The clearance of flight control law for a hypersonic gliding vehicle (HGV) and two linear clearance criteria based on structural singular value (μ) theory were proposed in [16]. However, since the aircraft is a nonlinear system, of which the mathematical model has parametric uncertainties, it is straightforward to deviate from the actual control by using direct linearization method.

Assume that the system is in instantaneous equilibrium, basic formula of linearized equations is then applied, and small deviation model of simplified equations is achieved [17]. In order to consider the effects of nonlinearity on the system, the article [18] shows the identifiability of a nonlinear delayed-differential model describing aircraft dynamics. In order to reduce the impact of model parameters perturbation on the system, a mixed $H2/H\infty$ control was proposed using fuzzy singularly perturbed model with multiple perturbation parameters [19]. A new strategy for missile attitude control using a hybridization of Linear Quadratic Gaussian (LQG), Loop Transfer Recovery (LTR), and Linear Quadratic Integral (LQI) control techniques was established [20]. However, it will result in relatively conservative results and will undermine the performance robustness of the system, due to the robust LQG control to maintain the minimum performance index. Guaranteed cost control (GCC) on uncertainty system is an effective method to solve the flaws of LQG design [21].

The GCC method can maintain the stability of the closed-loop system particularly when the controlled object has significant uncertainty. Meanwhile, it also ensures that the secondary performance index does not exceed the upper bound. A typical application of GCC method for a flexible air-breathing hypersonic vehicle (FAHV) can be found in [22]. In [23], the tracking GCC law was presented combined with the decoupling control to accommodate the parameter uncertainties without coupling. A modified GCC strategy has also been established for discrete-time uncertain systems with both state and input delays [24]. A robust guaranteed cost controller was proposed for quadrotor UAV system with uncertainties to address set-point tracking problem [25]. In order to eliminate disturbance effects and guarantee the robust stability of a quadrotor helicopter with state delay, improved guaranteed cost control and quantum adaptive control were developed [26]. A neural network (NN) based approximate optimal GCC design was developed to find a robust state feedback controller such that the closed-loop system has not only a bounded response in a finite duration of time for all admissible uncertainties but also a minimal guaranteed cost [27].

However, in order to obtain stronger robustness, robust control gains might be sensitive or fragile with respect to some errors or variations in control gains of feedback control. Therefore, a concept of nonfragile control strategy has been proposed, which gives a state feedback controller with enough regulating margin when control gains are varied. In [28], a synchronization problem for complex dynamical networks with additive time-varying coupling delays via non-fragile control was investigated. It has also been concerned with a problem of nonfragile robust optimal guaranteed cost control for a class of uncertain two-dimensional discrete state-delayed systems and the state feedback controllers are designed [29]. Robust nonfragile control of uncertain linear system and application to vehicle active suspension were described in [30]. In [31, 32], nonfragile guaranteed cost control of parametric uncertain systems was studied and the guaranteed cost nonfragile tracking control on the omnidirectional rehabilitative training walker was examined.

Though numerous research methods have been proposed in robust controller design, very limited work has been focused on the application of the hypersonic vehicles. Aiming at the complex hypersonic vehicle nonlinear system, small deviation linear equations are widely used in numerical analysis, but it may lead to the reduced model which can hardly achieve sufficient effect in the application of nonlinear system. The controller remains to be adjusted with considerable efforts before it can guarantee required control index. In this paper, we for the first time propose a linear and nonlinear combination in the course of system modeling, in order to make the expected model closer to the actual system. The Lyapunov function can be applied to get control law for nonlinear system when it satisfies certain Lipschitz conditions, and the problem is transformed into a feasible solution with linear matrix inequalities (LMI) method. Besides, adaptive SHLNN based nonfragile guaranteed cost control strategy is utilized to design the robust controller, with equivalent solution derived from LMI optimization approach. SHLNN are exploited as additive gain adjustments to eliminate the influence of results show that conservative control gains and counteract excessive upper bound of cost function are caused by uncertainties.

The rest of this article is organized as follows. In Section 2, motion model of hypersonic vehicle is formulated, where the state equations of vehicle body are established to testify the effectiveness of the proposed RGCC method. To prove the following theorems, several lemmas and assumptions are described in Section 3. Section 4 demonstrates the robust GCC law in the form of theorem under the Lipschitz conditions. In Section 5, a new adaptive nonfragile robust control strategy is presented, in which a nonfragile guaranteed cost controller solved by LMI optimization approach is applied. In Section 6, SHLNN controller design for nonfragile GCC strategy is treated as an additive gain compensator to reduce excessive performance caused by perturbations and uncertainties. Finally, simulation results of robust control and attitude tracking control are conducted and better stability and tracking performance by the proposed strategies for hypersonic vehicle model are gained.

2. Kinematics Model of Hypersonic Vehicle

According to the instantaneous equilibrium condition, the small deviation model of vehicle can be obtained based on the basic formulation of linearization equations. By analyzing motion mechanism and flight characteristics of the hypersonic vehicle, the motion equations of the vehicle body coordinate system are achieved as follows:

$$\dot{\alpha} = a_{11}F_{x1} + a_{12}F_{y1} + a_{13}\omega_{x1} + a_{14}\omega_{y1} + a_{15}\omega_{z1},$$

$$\dot{\beta} = b_{11}F_{x1} + b_{12}F_{y1} + b_{13}F_{z1} + b_{14}\omega_{x1} + b_{15}\omega_{y1},$$

$$\dot{\gamma} = c_{11}\omega_{x1} + c_{12}\omega_{y1} + c_{13}\omega_{z1},$$

$$J_{x1} \frac{d\omega_{x1}}{dt} + (J_{z1} - J_{y1}) \omega_{z1} \omega_{y1} = M_{x1},$$

$$\begin{aligned}
J_{y_1} \frac{d\omega_{y_1}}{dt} + (J_{x_1} - J_{z_1}) \omega_{x_1} \omega_{z_1} &= M_{y_1}, \\
J_{z_1} \frac{d\omega_{z_1}}{dt} + (J_{y_1} - J_{x_1}) \omega_{y_1} \omega_{x_1} &= M_{z_1},
\end{aligned} \tag{1}$$

where α represents attack angle, β denotes sideslip angle, and γ is roll angle. $F_{x_1}, F_{y_1}, F_{z_1}$ represent the components acting on missile body coordinates; $\omega_{x_1}, \omega_{y_1}, \omega_{z_1}$ represent the ω on x -, y -, z -axis of missile body coordinates. $J_{x_1}, J_{y_1}, J_{z_1}$ are vehicle's moment of inertias relative to each axis of vehicle body coordinate system; $d\omega_{x_1}/dt, d\omega_{y_1}/dt, d\omega_{z_1}/dt$ are components of vehicle rotation angular acceleration vector on each axis, respectively.

In these equations, values of parameters $a, b,$ and c are varied in aerodynamic model of vehicle. During the entire flight course, dramatic environmental changes will cause tens or even hundreds of times change of the aerodynamic parameters, which results in significant uncertainties on the mathematical vehicle model.

3. State Equation Description Form of Hypersonic Vehicle

Aiming at a classical nonlinear uncertain system, the state equation can be described as follows:

$$\begin{aligned}
\dot{x}(t) &= (A_1 + \Delta A)x(t) + (B_1 + \Delta B)u(t) + f(x, t), \\
y(t) &= Cx(t).
\end{aligned} \tag{2}$$

$x(t) \in R^n$ represents state vector, and $x(0) = x_0, u(t) \in R^m$ is control input vector, and $f(x, t) \in R^n$ is a nonlinear part and is a state-related nonlinear function which meets the global Lipschitz condition in Assumption 5. A_1 and B_1 are matrices with the certain dimension. ΔA and ΔB represent parameter uncertainties, assuming that the uncertainties are norm-bounded, which can be expressed as follows:

$$[\Delta A \quad \Delta B] = DF(t) [E_1 \quad E_2], \tag{3}$$

where $D \in R^{n \times r}, E_1 \in R^{q \times n}, E_2 \in R^{q \times m}$ are known real matrices with specific dimension, which characterize the structure of uncertainty in the system, and $F(t) \in R^{r \times q}$ is an unknown time-varying matrix, but norm-bounded as follows.

$$\Omega = \{F(t) \mid F^T(t)F(t) \leq I, \forall t\}. \tag{4}$$

The performance indicator is defined as follows.

$$J = \int_0^\infty [x^T(t)Qx(t) + u^T(t)Ru(t)] dt, \tag{5}$$

where Q and R are symmetric positive definite weighted matrices.

Lemma 1 (see [33]). For a given symmetric matrix, $F \in R^{n \times n}$ is expressed as

$$F = \begin{bmatrix} F_{11} & F_{12} \\ F_{21} & F_{22} \end{bmatrix}, \tag{6}$$

where $F_{11} \in R^{r \times r}, F_{12} \in R^{r \times (n-r)}, F_{21} \in R^{(n-r) \times r}, F_{22} \in R^{(n-r) \times (n-r)}$, and the conclusions are as follows.

- (1) $F < 0$;
- (2) $F_{11} < 0, F_{22} - F_{12}^T F_{11}^{-1} F_{12} < 0$;
- (3) $F_{22} < 0, F_{11} - F_{12} F_{22}^{-1} F_{12}^T < 0$.

Lemma 2. For $\sigma_1(y) = y^T Q_1 y \geq 0$, assuming there is $\tilde{y} \in R^m$, where $\sigma(\tilde{y}) > 0$, then the equivalent forms are as follows.

- (1) $y \in R^m$ makes $\sigma_1(y) \geq 0, y^T Q_0 y > 0$.
- (2) $\tau \geq 0$ makes $Q_0 - \tau Q_1 > 0$.

Corollary 3. When $P > 0$ and all $\xi \neq 0, \pi$ satisfying $\pi^T \pi \leq \xi^T C^T C \xi$ is established.

$$\begin{bmatrix} \xi \\ \pi \end{bmatrix}^T \begin{bmatrix} A^T P + PA & PB \\ B^T P & 0 \end{bmatrix} \begin{bmatrix} \xi \\ \pi \end{bmatrix} < 0. \tag{7}$$

When $\tau \geq 0$ and $P > 0$, then

$$\begin{bmatrix} A^T P + PA + \tau C^T C & PB \\ B^T P & -\tau I \end{bmatrix} < 0. \tag{8}$$

Lemma 4. When $D, E,$ and $F(t)$ satisfy the certain dimension real matrices, and $F^T(t)F(t) \leq I$, one can get the inequality for $\varepsilon > 0$.

$$DF(t)E + E^T F^T(t)D^T \leq \varepsilon DD^T + \varepsilon^{-1} E^T E. \tag{9}$$

Assumption 5. Nonlinear function $f(x, t)$ meets global Lipschitz condition, namely,

$$\begin{aligned}
\|f(x, t)\| &\leq \|Gx(t)\|, \\
\|f(x, t) - f(y, t)\| &\leq \|G(x(t) - y(t))\|.
\end{aligned} \tag{10}$$

4. Robust Guaranteed Cost Control of Hypersonic Vehicle

Theorem 6. According to the parameter uncertain system (2), if $f(x, t) = 0$ and it meets the performance (5), then there exists $u(t) = Kx(t)$ which satisfies the sufficient and necessary conditions for robust guaranteed cost with parameter uncertain closed-loop system: (1) there exists an appropriate constant $\varepsilon > 0$, which makes inequality (11) have a positive definite solution $P > 0$; (2) the robustness performance index of closed-loop system meets $J \leq \text{tr}(P)$ at the same time.

$$\begin{aligned}
&(A_1 + B_1 K + \Delta A + \Delta BK)^T P \\
&+ P(A_1 + B_1 K + \Delta A + \Delta BK) + Q + K^T R K < 0.
\end{aligned} \tag{11}$$

Corollary 7. Given that formula (11) is satisfied, there exist P and X which establish the existence of appropriate positive

constant, and matrices W and X satisfy allowable uncertainties [34].

$$\begin{bmatrix} \Pi & \varepsilon D & (E_1 X + E_2 W)^T & X & W^T \\ * & -\varepsilon I & 0 & 0 & 0 \\ * & * & -\varepsilon I & 0 & 0 \\ * & * & * & -Q^{-1} & 0 \\ * & * & * & * & -R^{-1} \end{bmatrix} < 0, \quad (12)$$

where $\Pi = (A_1 X + B_1 W)^T + (A_1 X + B_1 W)$. “*” denotes the transpose of symmetric part in equalities, and the definitions in the following matrix are the same. Furthermore, if inequality (12) has a solution (W, X) , it can be described as follows:

$$u^*(t) = WX^{-1}x(t). \quad (13)$$

This denotes a RGCC law of vehicle system. The performance indicator upper bound is

$$\bar{J} \leq \text{Trace}(X^{-1}) = \bar{J}^*. \quad (14)$$

Proof. For system (2), order $f(x, t) = 0$, and based on Lemma 4 and (3), inequality (11) can be transformed into

$$\begin{aligned} & (A_1 + B_1 K)^T P + P(A_1 + B_1 K) + \varepsilon P D D^T P \\ & + \varepsilon^{-1} (E_1 + E_2 K)^T (E_1 + E_2 K) + Q + K^T R K \quad (15) \\ & < 0. \end{aligned}$$

For inequality (15), based on Lemma 1, the following linear matrix inequalities can be obtained.

$$\begin{bmatrix} \Pi_1 & (E_1 + E_2 K)^T & I & K^T \\ * & -\varepsilon I & 0 & 0 \\ * & * & -Q^{-1} & 0 \\ * & * & * & -R^{-1} \end{bmatrix} < 0. \quad (16)$$

For inequality (16), multiply it by $\text{diag}\{P^{-1}, I, I, I\}$, and let $P^{-1} = X$, $KX = W$; then inequality (12) is obtained based on Lemma 1. We introduce the equation of $V(x(t)) = x^T(t)Px(t)$; then formula (17) for uncertain closed-loop system is obtained:

$$\begin{aligned} \dot{V}(x(t)) &= \dot{x}^T(t)Px(t) + x^T(t)P\dot{x}(t) = x^T(A_1 \\ &+ B_1 K + \Delta A + \Delta BK)^T Px(t) + x^T P(A_1 + B_1 K \\ &+ \Delta A + \Delta BK)x(t) \quad (17) \\ &= x^T[(A_1 + B_1 K + \Delta A + \Delta BK)^T P \\ &+ P(A_1 + B_1 K + \Delta A + \Delta BK)]x(t). \end{aligned}$$

We know from Theorem 6 that

$$\begin{aligned} \dot{V}(x(t)) &< -x^T(t)(Q + K^T R K)x(t) \\ \int_0^\infty \dot{V}(x(t)) dt &= V(x(\infty)) - V(x(0)) \quad (18) \\ &< -\int_0^\infty x^T(t)(Q + K^T R K)x(t) dt. \end{aligned}$$

On the basis of the stability condition of system, we get $V(x(\infty)) = 0$; then

$$\begin{aligned} \int_0^\infty x^T(t)(Q + K^T R K)x(t) dt &< V(x(0)) \quad (19) \\ &= x^T(0)Px(0). \end{aligned}$$

Thereupon we get in a further way

$$\bar{J} \leq E\{V(x(0))\} \text{Trace}(X^{-1}) = \bar{J}^*. \quad (20)$$

Proof is over. \square

Theorem 8. Aiming at uncertain nonlinear system (2) as well as index (5), if there are matrices X, Y and quantity $\varepsilon > 0$, $\tau > 0$, the following inequality will hold:

$$\begin{bmatrix} \Theta_1 & \tau I & \varepsilon D & \Theta_2 & XQ & W^T R & XG^T \\ * & -\tau I & 0 & 0 & 0 & 0 & 0 \\ * & * & -\varepsilon I & 0 & 0 & 0 & 0 \\ * & * & * & -\varepsilon I & 0 & 0 & 0 \\ * & * & * & * & -Q & 0 & 0 \\ * & * & * & * & * & -R & 0 \\ * & * & * & * & * & * & -\tau I \end{bmatrix} < 0, \quad (21)$$

where $\Theta_1 = A_1 X + X A_1^T + B_1 W + W^T B_1^T$, $\Theta_2 = (E_1 X + E_2 W)^T$.

Then $u = Kx(t)$ is RGCC control law of system (2), where $K = WX^{-1}$, and performance index is $J^* \leq x^T(0)X^{-1}x(0)$.

Proof. Considering function $V(x(t)) = x^T(t)Px(t)$, we take $u = Kx(t)$ into (2):

$$\begin{aligned} \dot{V}(x(t)) &+ x^T(t)Qx(t) + x^T(t)K^T R Kx(t) = \dot{x}^T(t) \\ &\cdot Px(t) + x^T(t)P\dot{x}(t) + x^T(t)Qx(t) + x^T(t) \\ &\cdot K^T R Kx(t) = [x(t)(A_1 + B_1 K + \Delta A + \Delta BK) \\ &+ f^T(x, t)]Px(t) + x^T(t) \\ &\cdot P[(A_1 + B_1 K + \Delta A + \Delta BK)x(t) + f(x, t)] \\ &+ x^T(t)Qx(t) + x^T(t)K^T R Kx(t) = x^T(t) \\ &\cdot [(A_1 + B_1 K + \Delta A + \Delta BK)^T P \end{aligned}$$

$$\begin{aligned}
& + P(A_1 + B_1K + \Delta A + \Delta BK)]x(t) + x^T(t) \\
& \cdot Pf(x,t) + f^T(x,t)Px(t) + x^T(t)Qx(t) \\
& + x^T(t)K^TRKx(t).
\end{aligned} \tag{22}$$

Order $\zeta^T(t) = [x^T(t) \ f^T(x,t)]$; then

$$\begin{aligned}
\dot{V}(x(t)) + x^T(t)Qx(t) + x^T(t)K^TRKx(t) \\
= z^T(t) \begin{pmatrix} \Omega & P \\ P & 0 \end{pmatrix} z(t),
\end{aligned} \tag{23}$$

where $\Omega = (A_1 + B_1K + \Delta A + \Delta BK)^TP + P(A_1 + B_1K + \Delta A + \Delta BK) + Q + K^TRK$.

Based on Assumption 5 we get

$$\zeta^T(t) \begin{bmatrix} -G^TG & 0 \\ 0 & I \end{bmatrix} \zeta(t) \leq 0. \tag{24}$$

Considering Lemma 2 and Corollary 3, when $\tau_0 > 0$, we can get

$$\begin{bmatrix} \Omega & P \\ P & 0 \end{bmatrix} - \tau_0 \begin{bmatrix} -G^TG & 0 \\ 0 & I \end{bmatrix} < 0. \tag{25}$$

Then,

$$\dot{V}(x(t)) + x^T(t)Qx(t) + x^T(t)K^TRKx(t) < 0. \tag{26}$$

Namely,

$$\begin{bmatrix} \Omega + \tau_0 G^TG & P \\ P & -\tau_0 I \end{bmatrix} < 0. \tag{27}$$

Multiplying $\text{diag}(P^{-1} \ I)$ on inequality (27) left and right, we have

$$\begin{bmatrix} P^{-1}(\Omega + \tau_0 G^TG)P^{-1} & I \\ I & -\tau_0 I \end{bmatrix} < 0. \tag{28}$$

Multiplying inequality (28) by $\text{diag}(I \ \tau_0^{-1}I)$ from both sides, we get

$$\begin{bmatrix} P^{-1}(\Omega + \tau_0 G^TG)P^{-1} & \tau_0^{-1}I \\ \tau_0^{-1}I & -\tau_0^{-1}I \end{bmatrix} < 0. \tag{29}$$

Order $P^{-1} = X$, $KX = W$, and $\tau_0^{-1} = \tau$, and based on Lemma 4, we get

$$\begin{bmatrix} \Xi & \tau I \\ \tau I & -\tau I \end{bmatrix} < 0, \tag{30}$$

where

$$\begin{aligned}
\Xi & = A_1X + XA_1^T + B_1W + W^TB_1^T + \varepsilon_1DD^T + \varepsilon_2DD^T \\
& + \varepsilon_1^{-1}(E_1X)^T(E_1X) + \varepsilon_2^{-1}(E_2W)^T(E_2W) \\
& + XQX + W^TRW + \tau^{-1}XG^TGX.
\end{aligned} \tag{31}$$

Based on Lemma 1 we know that (30) and (21) are equivalent. The proof is finished. Then from (26), we get

$$\dot{V}(x(t)) < -x^T(t)(Q + K^TRK)x(t) < 0. \tag{32} \quad \square$$

Under this condition, the system is stable.

Integrating both sides of formula (32), we have

$$\begin{aligned}
\int_0^\infty \dot{V}(x(t)) dt & = V(x(\infty)) - V(x(0)) \\
& < - \int_0^\infty x^T(t)(Q + K^TRK)x(t) dt.
\end{aligned} \tag{33}$$

According to system stability conditions, $V(x(\infty)) = 0$; then

$$\begin{aligned}
\int_0^\infty x^T(t)(Q + K^TRK)x(t) dt & < V(x(0)) \\
& = x^T(0)Px(0).
\end{aligned} \tag{34}$$

That is,

$$J^* \leq E\{V(x(0))\} = E\{x^T(0)Px(0)\} = \text{tr}(X^{-1}). \tag{35}$$

Theorem 9. Towards uncertain system (2) as well as performance index (5), the following optimization problem

$$\begin{aligned}
& \min_{\varepsilon, \tau, X, Y, M} \text{Trace}(M) \\
& \begin{bmatrix} \Theta_1 & \tau I & \varepsilon D & \Theta_2 & XQ & W^TR & XG^T \\ * & -\tau I & 0 & 0 & 0 & 0 & 0 \\ * & * & -\varepsilon I & 0 & 0 & 0 & 0 \\ * & * & * & -\varepsilon I & 0 & 0 & 0 \\ * & * & * & * & -Q & 0 & 0 \\ * & * & * & * & * & -R & 0 \\ * & * & * & * & * & * & -\tau I \end{bmatrix} < 0, \\
& \begin{bmatrix} M & I \\ * & X \end{bmatrix} > 0
\end{aligned} \tag{36}$$

has a solution $(\bar{\varepsilon}, \bar{\tau}, \bar{X}, \bar{W}, \bar{M})$; then $u^*(t) = \bar{W}\bar{X}^{-1}x(t)$ will be the optimal state feedback GCC law for such system. More free variables were introduced into the problem above, so that the solution to (36) was less conservative.

5. Nonfragile Guaranteed Cost Control Containing Nonlinear Perturbation

In order to reduce the system tracking error, we suppose that the output y_{dr} of system is constant vector which is a nonzero constant vector, and then the error vector is $e = y(t) - y_{dr} = x(t) - y_{dr}$. So the error system can be deduced such that

$$\begin{aligned}
\dot{e} & = (A_1 + \Delta A)e + (B_1 + \Delta B)u_k + (A_1 + \Delta A)y_{dr} \\
& + f_L(x, t).
\end{aligned} \tag{37}$$

For formula (37), design controller is with guaranteed cost. Consider that reference state input y_{dr} is bounded and assume that nonlinear function $f_L(e, \xi, t) = f_L(x(t))$ satisfies

$$f_L^T(e, \xi, t) f_L(e, \xi, t) \leq e^T(t) G^T G e(t), \quad (38)$$

where G is a constant matrix. Meanwhile, it satisfies $G^T G > 0$.

It is clear that the regulator of system (37) is equal to design of the tracking controller for system (2). Let $u(t) = u_{ke} + v_{tr} = Ke + v_{tr}$, and (37) can be expressed as

$$\begin{aligned} \dot{e} = & ((A_1 + \Delta A) + (B_1 + \Delta B)K)e + (B_1 + \Delta B)v_{tr} \\ & + (A_1 + \Delta A)y_{dr} + f_L(e, \xi, t). \end{aligned} \quad (39)$$

To realize the regulation and control of system (37), it is requested that 0 is the balance point of this system; thus let

$$(B_1 + \Delta B)v_{tr} + (A_1 + \Delta A)y_{dr} + f_L(e, \xi, t) = 0. \quad (40)$$

If this system is progressively stable, then u_∞ is approximated to meet formula (41).

When $t \rightarrow \infty$, $e(t) \rightarrow 0$, $\dot{e}(t) \rightarrow 0$, $f_L(e, \xi, t) \rightarrow 0$, $u(\infty) \rightarrow u_\infty = v_{tr}$, then get

$$(B_1 + \Delta B)u_\infty + (A_1 + \Delta A)y_{dr} = 0. \quad (41)$$

That is,

$$v_{tr} = u_\infty = -B_1^+ A_1 y_{dr}. \quad (42)$$

Let $K_c = -B_1^+ A_1$; a feedback controller with uncertainties is given as

$$u(t) = v_k + v_{tr} = (K + \Delta K)e(t) + K_c y_{dr}, \quad (43)$$

where K is state feedback matrix and K_c is feedforward compensation matrix. ΔK is an uncertain matrix with corresponding dimension, which is norm-bounded in assumption and satisfies

$$\Delta K = D_K N_K(t) E_K, \quad (44)$$

where D_K, E_K are known matrices and $N_K(t)$ is an unknown time-varying matrix, represents a neural network control output, and satisfies

$$N_K^T(t) N_K(t) \leq I. \quad (45)$$

Order $u_e(t) = u(t) - u_\infty$, and define quadratic performance index as the tracking performance of system; we have

$$J_e = \int_0^\infty (e^T(t) Q e(t) + u_e^T(t) R u_e(t)) dt. \quad (46)$$

Definition 10. For uncertain system (37) and cost function (46), K^* can be defined as a nonfragile guaranteed cost control gain matrix with the corresponding upper bound J^* of cost function, only if there exists a controller (43) satisfying inequality (45), which makes the closed-loop system with system uncertainties in (3) and nonlinear perturbation in (38) asymptotically stable, where K^* is a constant gain matrix and $J^* \geq J$ is a positive constant.

Theorem 11. For an uncertain system (37) and cost function (46), if there exist symmetric positive definite matrices P and K , with a scalar quantity $\varepsilon_1 > 0$, satisfying the following equation

$$M = \begin{bmatrix} Q + \bar{K}^T R \bar{K} + \Lambda_1 + \Lambda_1^T + \Lambda_2 & P \\ P & -\varepsilon_1^{-1} I \end{bmatrix} < 0, \quad (47)$$

K is a nonfragile guaranteed cost control gain matrix and $J^* = e_0^T P e_0$ is the upper bound of cost function (46), where

$$\begin{aligned} \Lambda_1 &= P(\bar{A} + \bar{B}\bar{K}), \\ \Lambda_2 &= \varepsilon_1^{-1} G^T G, \\ \bar{A} &= A + \Delta A, \\ \bar{B} &= B + \Delta B, \\ \bar{K} &= K + \Delta K. \end{aligned} \quad (48)$$

Proof. Select the Lyapunov function $V(t) = e^T(t) P e(t)$, where P is a positive definite matrix, and based on the control law (43), the time derivative of $V(t)$ with respect to time t yields [35]

$$\dot{V}(e) = \begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix}^T \begin{bmatrix} \Lambda_1^T + \Lambda_1 & P \\ P & 0 \end{bmatrix} \begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix}. \quad (49)$$

And (37) can be transformed into

$$\begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix}^T \begin{bmatrix} -G^T G & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix} < 0. \quad (50)$$

According to matrix inequality (47), it can be concluded that

$$\begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix}^T M \begin{bmatrix} e(t) \\ f_L(e, v, t) \end{bmatrix} < 0. \quad (51)$$

Multiply ε_1 and add (51) to the left side of (50); it can be derived as $\dot{V}(e) < -e^T Q e - e^T \bar{K}^T R \bar{K} e = -e^T Q e - v_k^T R v_k < 0$. According to Lyapunov stability theory, system (37) is asymptotically stable.

After the integration of (51) on both sides from $t = 0$ to $t = \infty$ and equation $e(\infty) = 0$ inferred from asymptotic stability of the closed-loop system, it can be concluded that

$$\begin{aligned} J &= \int_0^\infty e^T (Q + \bar{K}^T R \bar{K}) e dt < V(e(0)) - V(e(\infty)) \\ &= e_0^T P e_0 = J^*. \end{aligned} \quad (52)$$

This completes the proof. \square

In order to carry out nonfragile guaranteed cost controller for the system, the equivalent LMI expression of condition (45) is given based on Theorem 12.

Theorem 12. For given positive definite matrices P and K , the closed-loop system (37) has a feasible solution $(\rho, \varepsilon_1, \varepsilon_2, Y, X)$, which guarantees the establishment of condition (47) for all allowable uncertainty. A nonfragile state feedback controller

gain matrix $K = YX^{-1}$ and an upper bound cost function $\bar{J} \leq \text{Trace}(X^{-1}) = \bar{J}^*$ exist, if and only if there exist $\rho, \varepsilon_1, \varepsilon_2 > 0$, symmetric positive definite matrix X , and real matrix Y , such that the following LMI holds:

$$\begin{bmatrix} \Lambda + \rho DD^T & * & * & * & * & * & * & * \\ E_1 X + E_2 Y & -\rho I & * & * & * & * & * & * \\ X & 0 & -Q^{-1} & * & * & * & * & * \\ Y & 0 & 0 & -R^{-1} & * & * & * & * \\ E_K X & 0 & 0 & 0 & -\varepsilon_2 I & * & * & * \\ \varepsilon_2 D_K^T B^T & \varepsilon_2 D_K^T E_2^T & 0 & \varepsilon_2 D_K^T & 0 & -\varepsilon_2 I & * & * \\ X & 0 & 0 & 0 & 0 & 0 & -\varepsilon_1 (G^T G)^{-1} & * \\ \varepsilon_1 I & 0 & 0 & 0 & 0 & 0 & 0 & -\varepsilon_1 I \end{bmatrix} < 0, \quad (53)$$

where $\Lambda = AX + BY + (AX + BY)^T$ and $*$ is the corresponding symmetric part of the matrix.

Proof. Following proof process of Theorem 11, we can transform the existence condition (47) based on Lemmas 1 and 2 into the following expression:

$$\begin{bmatrix} \Lambda_2 + \Lambda_3 + \Lambda_4 + \Lambda_5 & * & * & * \\ E_1 + E_2 \tilde{K} & -\rho I & * & * \\ I & 0 & -Q^{-1} & * \\ \tilde{K} & 0 & 0 & -R^{-1} \end{bmatrix} < 0, \quad (54)$$

where $\Lambda_3 = P(A + B\tilde{K}) + (A + B\tilde{K})^T P$, $\Lambda_4 = \rho PDD^T P$, $\Lambda_5 = \varepsilon_1 P^T P$. Substituting (44) into left side of (54), it can be decomposed as

$$Y_1 + \Sigma_1 + \Sigma_1^T < 0, \quad (55)$$

where

$$Y_1 = \begin{bmatrix} \Lambda_2 + \Lambda_4 + G^T G & * & * & * \\ E_1 + E_2 K & -\rho I & * & * \\ I & 0 & -Q^{-1} & * \\ K & 0 & 0 & -R^{-1} \end{bmatrix}, \quad (56)$$

$$\Sigma_1 = \begin{bmatrix} PBD_K \\ E_2 D_K \\ 0 \\ D_K \end{bmatrix} N_K [E_K \ 0 \ 0 \ 0].$$

According to matrix inequality lemma, for all matrices N_K meeting $N_K^T N_K \leq I$ and a scalar $\varepsilon_2 > 0$, (55) is equivalent to

$$Y_1 + \varepsilon_2 \begin{bmatrix} PBD_K \\ E_2 D_K \\ 0 \\ D_K \end{bmatrix} \begin{bmatrix} PBD_K \\ E_2 D_K \\ 0 \\ D_K \end{bmatrix}^T + \varepsilon_2^{-1} \begin{bmatrix} E_K^T \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} E_K^T \\ 0 \\ 0 \\ 0 \end{bmatrix}^T < 0. \quad (57)$$

Applying Schur complement and multiplying each side of (57) by diag matrix $\{P^{-1}, I, I, I, I, \varepsilon_2 I, I, \varepsilon_1 I\}$, the linear matrix inequality (53) including variables $\rho, \varepsilon_1, \varepsilon_2$ and matrices Y and X can be obtained and parametric expression of the nonfragile guarantee cost control gain is given as $K = YX^{-1}$, where $X = P^{-1}$, $Y = KP^{-1}$. Proof is then finished. \square

Theorem 13. For (37) and cost function (46), there exist an optimal nonfragile guaranteed cost control gain matrix $K^* = Y^*(X^*)^{-1}$ and a minimal upper bound $\text{Trace}(S)$ of the cost function J^* , and the following optimization problem

$$\min_{\rho, \varepsilon_1, \varepsilon_2, Y, X, S} \text{Trace}(S) \quad (58)$$

has a solution as $(\rho^*, \varepsilon_1^*, \varepsilon_2^*, Y^*, X^*, S^*)$, satisfying both the linear matrix inequality (53) and the following equation:

$$\begin{bmatrix} S & * \\ I & X \end{bmatrix} > 0. \quad (59)$$

Proof. If $(\rho^*, \varepsilon_1^*, \varepsilon_2^*, Y^*, X^*, S^*)$ is a feasible solution of (58), it is also feasible to (53). According to Theorems 11 and 12, $K^* = Y^*(X^*)^{-1}$ is a feasible nonfragile guaranteed cost control gain matrix for the system. And using Lemma 1, (59) is equivalent to $S > X^{-1} > 0$, and the minimum of $\text{Trace}(S)$ will ensure the minimization of $\text{Trace}(X^{-1})$, which is the minimization of cost function's upper bound. This completes the proof. \square

7. Simulation Results Analysis

7.1. State Regulating Simulation of Hypersonic Vehicle. For a specific hypersonic vehicle system, the relevant parameters of the system equation are given as follows.

$$\begin{aligned}
 A_1 &= \begin{bmatrix} -0.0299 & 1.0000 & 0 & 0 & 0 & 0 \\ -0.6345 & -0.0184 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.0058 & 0.9879 & 0 & 0.1543 \\ 0 & 0 & -1.0467 & -0.0063 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0000 \\ -0.0009 & 0 & 56.5463 & 0 & 0 & -0.0167 \end{bmatrix}, \\
 B_1 &= \begin{bmatrix} -0.0008 & 0 & 0 \\ -4.3885 & 0 & 0 \\ 0 & -0.0001 & 0.0001 \\ 0 & -0.3703 & 1.9365 \\ 0 & 0 & 0 \\ 0 & 0.6356 & -27.4183 \end{bmatrix}, \\
 D &= \text{diag}(1 \ 1 \ 1 \ 1 \ 1 \ 1),
 \end{aligned} \tag{66}$$

$$\begin{aligned}
 E_1 &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0.0346 & 0.0001 & 0.0001 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.0001 & 0 & 0.0009 \\ 0.0001 & 0 & 0.0088 & 0 & 0 & 0 \\ 0 & 0 & 0.01 & 0 & 0 & 0 \\ 0.0012 & 0 & 0.2974 & 0 & 0 & 0.0001 \end{bmatrix}, \\
 E_2 &= \begin{bmatrix} 0 & 0 & 0 \\ 0.0177 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0.0015 & 0.0103 \\ 0 & 0 & 0 \\ 0 & 0.0025 & 0.1295 \end{bmatrix},
 \end{aligned}$$

$$Q = \text{diag}(20 \ 0.1 \ 20 \ 0.1 \ 20 \ 0.1),$$

$$R = \text{diag}(1 \ 1 \ 1).$$

According to Theorem 6, matrix K can be achieved.

$$\begin{aligned}
 K &= \begin{bmatrix} 7.3272 & 4.1847 & 0.0280 & 0.0079 & -0.0012 & 0.0007 \\ 0.0091 & 0.0005 & 40.3491 & 19.6845 & -2.7003 & 1.0848 \\ 0.0009 & 0.0002 & 4.3130 & 0.9314 & 3.2004 & 1.7965 \end{bmatrix}. \tag{67}
 \end{aligned}$$

From the above equations, we can get the performance index $J^* = 4.0560$ through Matlab simulation. Given the initial state vector $\mathbf{X}_0 = [0.2 \ 0.1 \ 0.1 \ 0.1 \ 0.1 \ 0.1]^T$, the simulation results of $\mathbf{x}(t)$ and $u(t)$ are shown in Figures 2 and 3, respectively. From Figure 2, six response curves of

angles and angular velocities show that the regulating system is stable and controllable, with short settling time. Through the definition of $J(t) = \int_0^t [x^T(\tau)Qx(\tau) + u^T(\tau)Ru(\tau)]d\tau$, the evolution process of $J(t)$ is shown in Figure 4, which shows that the system has a given performance index upper bound.

7.2. Robust Tracking Control Simulation Result of Hypersonic Vehicle. In numerical simulation, we decompose ΔA and ΔB like Section 7.1. Suppose there exists $G^T G = 0.5I$ in (38) and number of SHLNN network outputs in the matrix N_K is three. Furthermore, in order to guarantee the ability to counteract system uncertainties, gain matrices D_K and E_K are given as

$$\begin{aligned}
 D_K &= \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}, \\
 E_K &= \begin{bmatrix} 20 & 15 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 20 & 15 \end{bmatrix}. \tag{68}
 \end{aligned}$$

So the gain matrix \mathbf{K} can be obtained by LMI optimization approach according to Theorems 12 and 13.

$$\begin{aligned}
 \mathbf{K} &= \begin{bmatrix} 29.7482 & 22.4213 & 0.0166 & 0.0044 & -0.0012 & 0.0004 \\ 0.0306 & 0.0004 & 74.1146 & 31.5391 & -8.0931 & -0.4972 \\ 0.0008 & 0.0000 & 3.9106 & 0.7396 & 11.1610 & 8.4109 \end{bmatrix}. \tag{69}
 \end{aligned}$$

Three SHLNN outputs are utilized to adjust gain coefficients of pitch, yaw, and roll, respectively. The network inputs are chosen as $\pi_\alpha = [\alpha_c, \dot{\alpha}_c, e_\alpha, u_\alpha]^T$, $\pi_\beta = [e_\beta, u_\beta]^T$, $\pi_\gamma = [\gamma_c, \dot{\gamma}_c, e_\gamma, u_\gamma]^T$, with predefined network outputs N_{Ki} . Moreover, learning rates in (65) are $\kappa_{W1} = \kappa_{V1} = \kappa_{W2} = \kappa_{V2} = 0.4$, $\kappa_{W3} = \kappa_{V3} = 0.2$, with inertial coefficients $\lambda_1 = \lambda_2 = \lambda_3 = 0.1$.

To test the tracking effect of the above control law, square waves are selected as the command of pitch and roll channels with the alternative amplitudes from 2.5° to 5° and -10° to 10° correspondingly, with time period as 20 seconds. In Figure 5, tracking performance results of GCC method and the proposed scheme are presented, where dash lines stand for standard GCC method results and solid lines are the responses using nonfragile robust control strategy. Figure 6 denotes the three channel angular velocity curves and Figure 7 is the curves of elevator and rudder angles in pitch, yaw, and roll channels. Figure 8 illustrates the regulating process of the control gain matrix $K + \Delta K$ elements, where ΔK is adjusted by the three SHLNN outputs. As shown above, the tracking effect of nonfragile robust control gives a good improvement in dynamic performance and tracking errors are apparently decreased. Therefore, the nonfragile guaranteed cost control method, integrated with SHLNN controller to update gain values, is effective in improving the control performance as proposed.

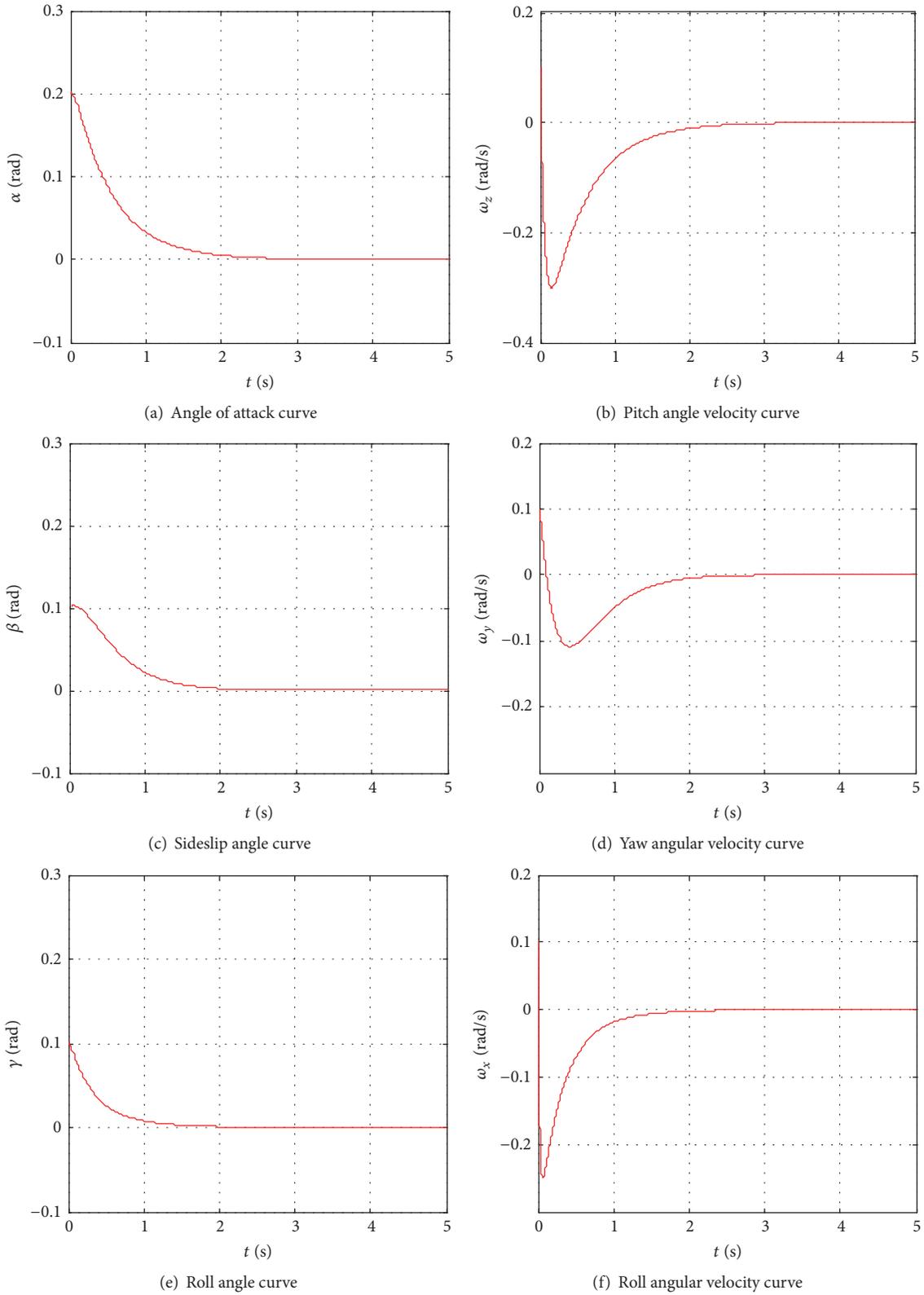


FIGURE 2: Responses of GCC controller.

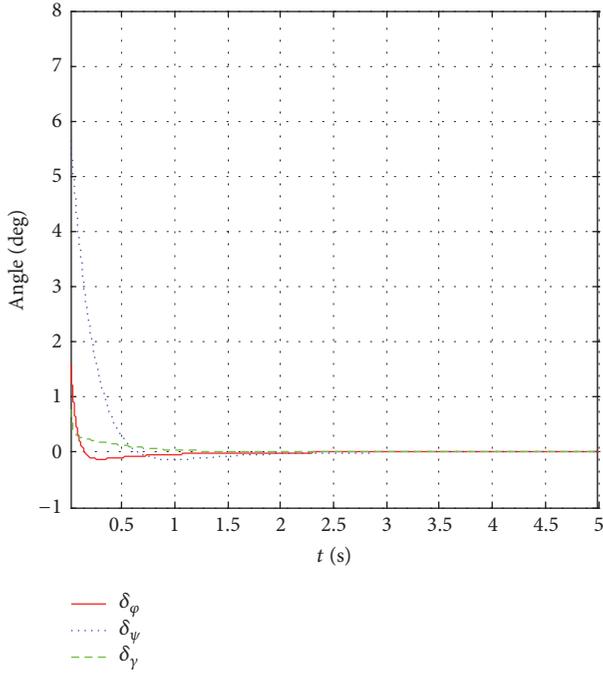


FIGURE 3: Regulating process of rudder angles as control inputs.

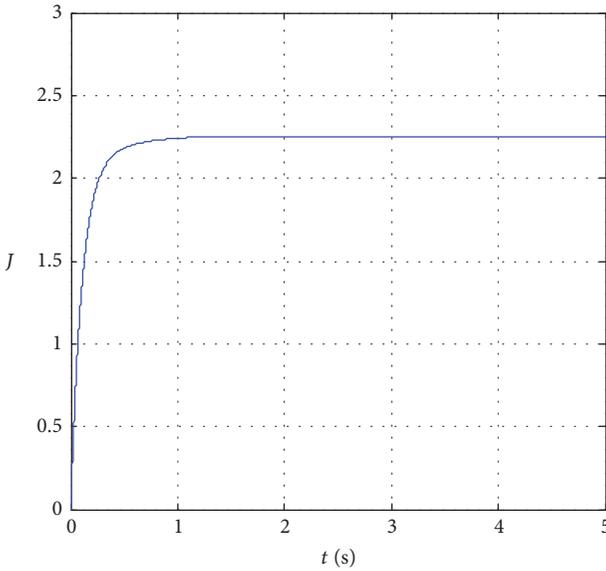
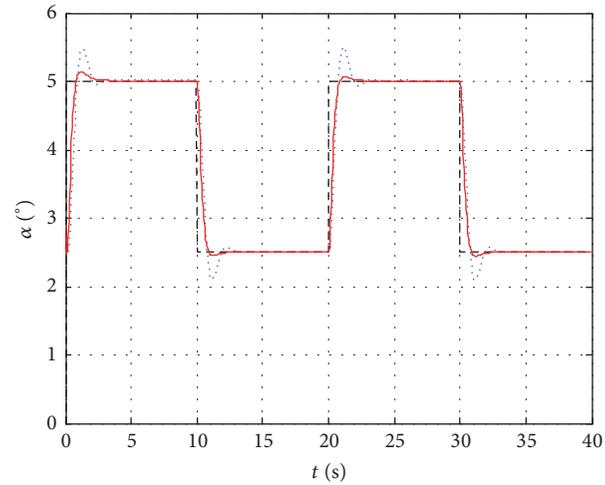


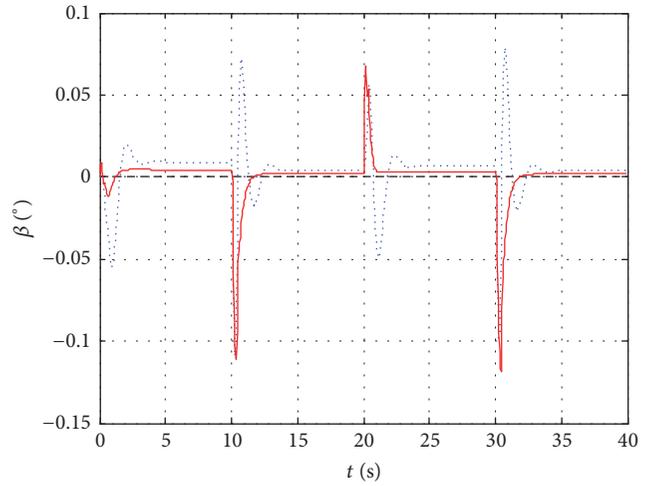
FIGURE 4: Performance index regulating process.

8. Conclusions

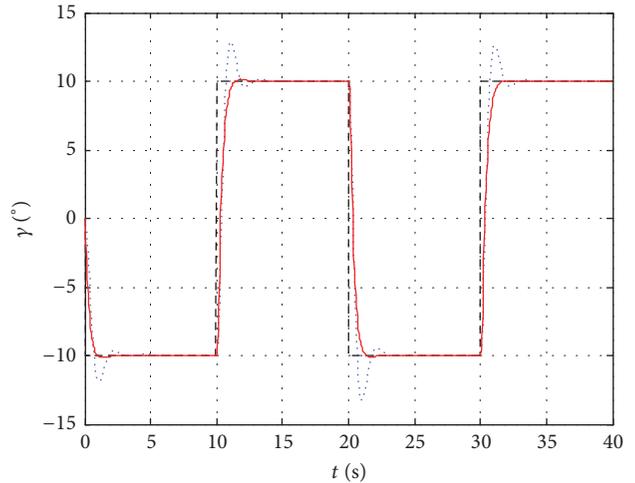
On the basis of RGCC design theory, the state feedback control law is obtained in this paper by applying Lyapunov function and satisfying overall Lipschitz condition. Specifically, the state-related nonlinear equation is built with a nonlinear part. The equations establishment formulates the process for solving the solution with LMI. Furthermore, adaptive SHLNN based nonfragile guaranteed cost control strategy is utilized to design the robust controller, with equivalent solution derived from LMI optimization approach,



(a) Tracking response of attack angle



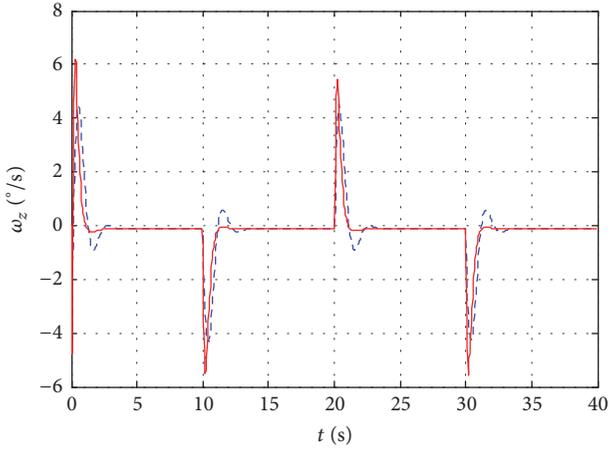
(b) Regulation response of sideslip angle



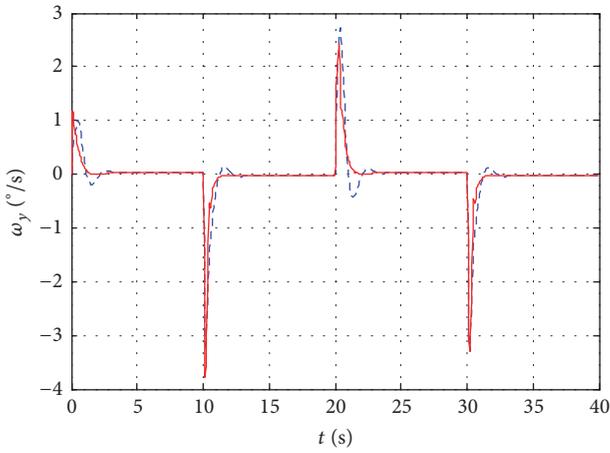
(c) Tracking response of roll angle

FIGURE 5: Tracking curves of three channel angles.

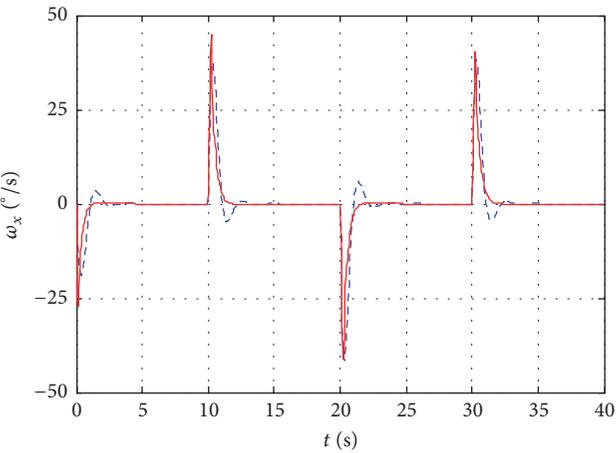
where SHLNN are exploited as additive gain adjustments to eliminate the influence of conservative control gains and



(a) Regulation response of pitch angular velocity

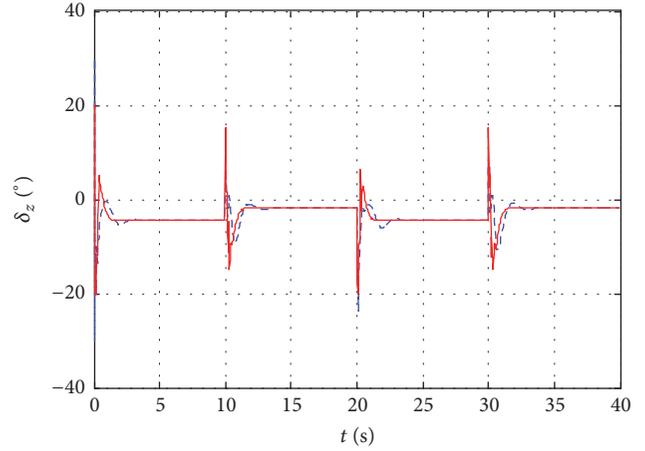


(b) Regulation response of yaw angular velocity

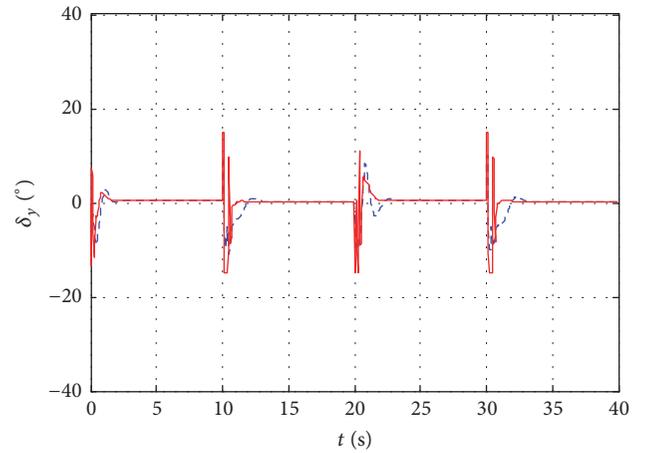


(c) Regulation response of roll angular velocity

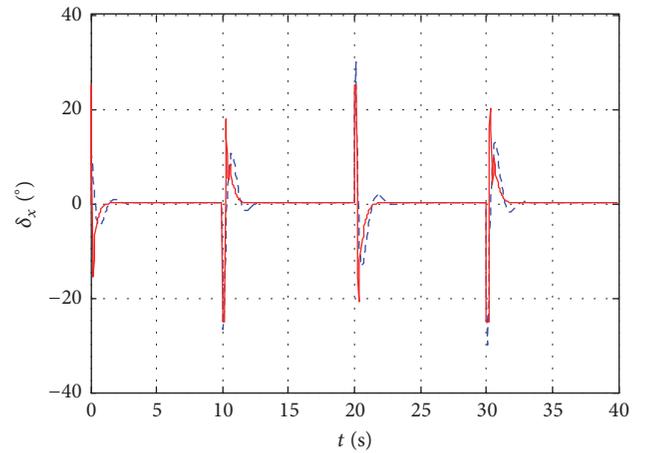
FIGURE 6: Regulation curves of angular velocities.



(a) Regulation curve of elevator angle in pitch channel



(b) Regulation curve of rudder angle in yaw channel

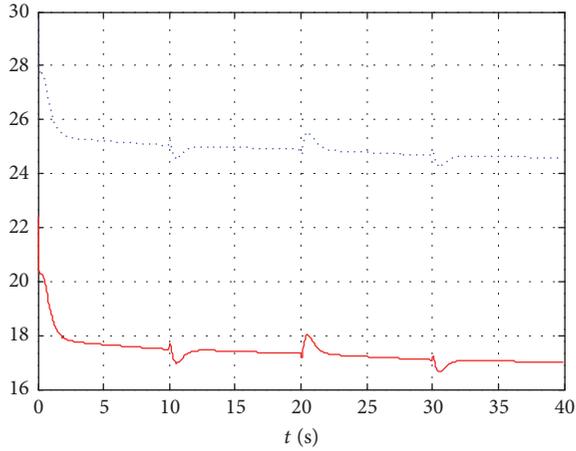


(c) Regulation curve of elevator angle in roll channel

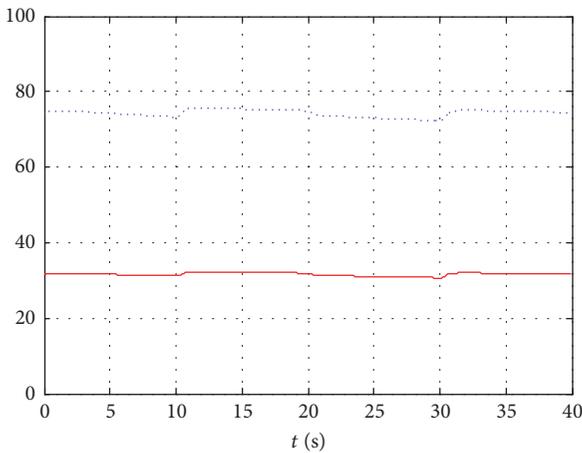
FIGURE 7: Regulation curves of elevator and rudder angles in three channels.

counteract excessive upper bound of cost function caused by uncertainties. Finally, simulation verifications are carried out with a specific model of hypersonic vehicle, and feasibility

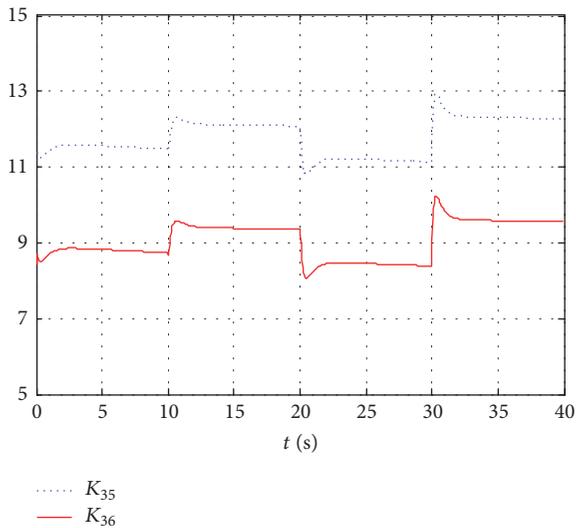
and adaptability of the proposed algorithm are demonstrated accordingly, where the proposed method has better tracking performance in attitude control on the vehicle.



(a) Gains variation in pitch channel



(b) Gains variation in yaw channel



(c) Gains variation in roll channel

FIGURE 8: Control gains variation in three channels.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (61473124, 61503173), the Science and Technology Research Project of Education Department of Henan Province (15A413016), and the Science and Technology Key Project of Henan Province (162102410051). This paper was also supported by the National Scholarship Fund.

References

- [1] R. A. Hess, "Frequency domain-based pseudosliding mode flight control design," *Journal of Aircraft*, vol. 49, no. 6, pp. 2077–2088, 2012.
- [2] H. Sun, S. Li, and C. Sun, "Finite time integral sliding mode control of hypersonic vehicles," *Nonlinear Dynamics*, vol. 73, no. 1-2, pp. 229–244, 2013.
- [3] D. Liu, C. Wu, Q. Zhou, and H.-K. Lam, "Fuzzy guaranteed cost output tracking control for fuzzy discrete-time systems with different premise variables," *Complexity*, vol. 21, no. 5, pp. 265–276, 2016.
- [4] Z. Jin, J. Chen, Y. Sheng, and X. Liu, "Neural network based adaptive fuzzy PID-type sliding mode attitude control for a reentry vehicle," *International Journal of Control, Automation, and Systems*, vol. 15, no. 1, pp. 404–415, 2017.
- [5] Y. Hou and S. Tong, "Adaptive fuzzy backstepping control for a class of MIMO switched nonlinear systems with unknown control directions," *Complexity*, vol. 21, no. 6, pp. 155–166, 2016.
- [6] J. Na, G. Herrmann, and K. Zhang, "Improving transient performance of adaptive control via a modified reference model and novel adaption," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 8, pp. 1351–1372, 2017.
- [7] D. D. Donalson and C. T. Leondes, "A model referenced parameter tracking technique for adaptive control systems: i—the principle of adaptation," *IEEE Transactions on Applications and Industry*, vol. 82, no. 68, pp. 241–252, 1963.
- [8] S. S. Dunn and R. Edelmann, "Minimum power spacecraft attitude control laws for small constant disturbance torques," *IEEE Transactions on Automatic Control*, vol. 13, no. 6, pp. 691–694, 1968.
- [9] G. Kreisselmeier and B. D. Anderson, "Robust model reference adaptive control," *Institute of Electrical and Electronics Engineers Transactions on Automatic Control*, vol. 31, no. 2, pp. 127–133, 1986.
- [10] L. Giovanini, "Robust adaptive control using multiple models, switching and tuning," *IET Control Theory and Applications*, vol. 5, no. 18, pp. 2168–2178, 2011.
- [11] Z. Pu, R. Yuan, X. Tan, and J. Yi, "Active robust control of uncertainty and flexibility suppression for air-breathing hypersonic vehicles," *Aerospace Science and Technology*, vol. 42, pp. 429–441, 2015.
- [12] J. Na, X. Ren, and D. Zheng, "Adaptive control for nonlinear pure-feedback systems with high-order sliding mode observer," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 3, pp. 370–382, 2013.

- [13] C. Peng, J. Zhang, and Y. C. Yan, "Adaptive event-triggering H_{∞} inf load frequency control for network-based power systems," *IEEE Transactions on Industrial Electronics*, no. 99, 2017.
- [14] R. Matušů and R. Prokop, "Computation of robustly stabilizing PID controllers for interval systems," *SpringerPlus*, vol. 5, no. 1, article no. 702, 2016.
- [15] E. Gershon, "Robust Reduced-order H-infinity Output-Feedback Control of Retarded Stochastic Linear Systems," *IEEE Transactions on Automatic Control*, vol. 58, no. 11, pp. 2898–2904, 2013.
- [16] L. Liu, S. Dong, Y. Wang, and L. Ou, "Clearance of flight control law based on structural singular value theory," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 2138–2147, 2015.
- [17] C. F. Li, Z. X. Ye, Y. J. Wang, and L. Liu, "Design of attitude decoupling control system for BTT missile using quantitative feedback theory," in *Proceedings of the 2010 International Conference on Modelling*, 2010.
- [18] L. Denis-Vidal, C. Jauberthie, and G. Joly-Blanchard, "Identifiability of a nonlinear delayed-differential aerospace model," *Institute of Electrical and Electronics Engineers Transactions on Automatic Control*, vol. 51, no. 1, pp. 154–158, 2006.
- [19] Y. Yuan, Y. Hu, and F. Sun, "Mixed H_2/H_{∞} control using a fuzzy singularly perturbed model with multiple perturbation parameters for gust load alleviation," *Tsinghua Science and Technology*, vol. 16, no. 4, pp. 344–351, 2011.
- [20] S. Das and K. Halder, "Missile attitude control via a hybrid LQG-LTR-LQI control scheme with optimum weight selection," in *Proceedings of the 1st International Conference on Automation, Control, Energy and Systems, ACES '14*, pp. 1–6, February 2014.
- [21] S. S. L. Chang and T. K. C. Peng, "Adaptive guaranteed cost control of systems with uncertain parameters," *IEEE Transactions on Automatic Control*, vol. 17, no. 4, pp. 474–483, 1972.
- [22] H. Li, Y. Si, L. Wu, X. Hu, and H. Gao, "Guaranteed cost control with poles assignment for a flexible air-breathing hypersonic vehicle," *International Journal of Systems Science*, vol. 42, no. 5, pp. 863–876, 2011.
- [23] Z. Deng, Y. Wang, L. Liu, and Q. Zhu, "Guaranteed cost decoupling control of bank-to-turn vehicle," *IET Control Theory & Applications*, vol. 4, no. 9, pp. 1594–1604, 2010.
- [24] E. Gyurkovics, "Guaranteed cost control of discrete-time uncertain systems with both state and input delays," *International Journal of Control*, vol. 89, no. 10, pp. 2073–2082, 2016.
- [25] Z. Xu, X. Nian, H. Wang, and Y. Chen, "Robust guaranteed cost tracking control of quadrotor UAV with uncertainties," *ISA Transactions*, vol. 69, pp. 157–165, 2017.
- [26] L. Hu, F. Chen, B. Jiang, and G. Tao, "Control strategy for a quadrotor helicopter with state delay via improved guaranteed cost control and quantum adaptive control," *Journal of Aerospace Engineering*, vol. 30, no. 4, 2017.
- [27] H.-N. Wu, M.-M. Li, and L. Guo, "Finite-horizon approximate optimal guaranteed cost control of uncertain nonlinear systems with application to Mars entry guidance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 7, pp. 1456–1467, 2015.
- [28] N. Sakthivel, R. Rakkiyappan, and J. H. Park, "Non-fragile synchronization control for complex networks with additive time-varying delays," *Complexity*, vol. 21, no. 1, pp. 296–321, 2015.
- [29] A. Tandon and A. Dhawan, "Non-fragile robust optimal guaranteed cost control of uncertain 2-D discrete state-delayed systems," *International Journal of Systems Science*, vol. 47, no. 14, pp. 3303–3319, 2016.
- [30] Y. Kong, D. Zhao, B. Yang, C. Han, and K. Han, "Robust non-fragile H-infinity/L-2-L-infinity control of uncertain linear system with time-delay and application to vehicle active suspension," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 13, pp. 2122–2141, 2015.
- [31] S. X. Guo, "Robust reliability method for non-fragile guaranteed cost control of parametric uncertain systems," *Systems and Control Letters*, vol. 64, pp. 27–35, 2014.
- [32] P. Sun and S. Wang, "Redundant input guaranteed cost non-fragile tracking control for omnidirectional rehabilitative training walker," *International Journal of Control, Automation, and Systems*, vol. 13, no. 2, pp. 454–462, 2015.
- [33] L. Yu, *Robust Control-Linear Matrix Inequality Method*, Tsinghua University Press, Beijing, China, 2002.
- [34] C. F. Li, L. Liu, Y. J. Wang, and Z. S. Wang, "Design of robust h-infinity control law via lmi for lifting vehicle," *Information-an International Interdisciplinary Journal*, vol. 15, no. 1, pp. 149–156, 2012.
- [35] C. Peng and J. Zhang, "Delay-distribution-dependent load frequency control of power systems with probabilistic interval delays," *IEEE Transactions on Power Systems*, vol. 31, no. 4, pp. 3309–3317, 2016.
- [36] J. Na, Q. Chen, X. Ren, and Y. Guo, "Adaptive prescribed performance motion control of servo mechanisms with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 486–494, 2014.
- [37] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, 2017.
- [38] G. Li, "Nonlinear model predictive control of a wave energy converter based on differential flatness parameterisation," *International Journal of Control*, vol. 90, no. 1, pp. 68–77, 2017.

Research Article

Intelligent Image Recognition System for Marine Fouling Using Softmax Transfer Learning and Deep Convolutional Neural Networks

C. S. Chin,¹ JianTing Si,^{1,2} A. S. Clare,³ and Maode Ma⁴

¹Faculty of Science, Agriculture and Engineering, Newcastle University Singapore, Newcastle upon Tyne, NE1 7RU, UK

²PACC Offshore Services Holdings (POSH) Ltd., No. 1 Kim Seng Promenade, No. 06-01 Great World City, Singapore 237994

³School of Natural and Environmental Sciences, Newcastle University Singapore, Newcastle upon Tyne NE1 7RU, UK

⁴School of Electrical & Electronic Engineering, Nanyang Technological University, Nanyang Avenue, Singapore 639798

Correspondence should be addressed to C. S. Chin; cheng.chin@ncl.ac.uk

Received 14 July 2017; Accepted 10 September 2017; Published 15 October 2017

Academic Editor: Jing Na

Copyright © 2017 C. S. Chin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The control of biofouling on marine vessels is challenging and costly. Early detection before hull performance is significantly affected is desirable, especially if “grooming” is an option. Here, a system is described to detect marine fouling at an early stage of development. In this study, an image of fouling can be transferred wirelessly via a mobile network for analysis. The proposed system utilizes transfer learning and deep convolutional neural network (CNN) to perform image recognition on the fouling image by classifying the detected fouling species and the density of fouling on the surface. Transfer learning using Google’s Inception V3 model with Softmax at last layer was carried out on a fouling database of 10 categories and 1825 images. Experimental results gave acceptable accuracies for fouling detection and recognition.

1. Introduction

Marine biofouling is the unwanted colonisation and growth of marine organisms on immersed artificial structures [1]. Following rapid conditioning of the surface, subsequent fouling is a dynamic process, which depends on availability, the relation of colonisers to incumbents, and the speed at which organisms can attach [2], though it is often described as successional [3]. With regard to shipping, the main issue for hull performance is an increase in frictional resistance, which requires increased power to maintain speed and hence increased fuel consumption and gaseous emissions [4, 5]. Hull fouling is also the main vector for translocation of nonindigenous species [6]. The early detection and recognition of marine growth are therefore paramount to avoid these problems. This paper studies marine growth detection and recognition using the transfer learning and deep convolutional neural network (CNN) approach.

Artificial Intelligence (AI) uses machine learning where the test data are extracted from a similar n -dimensional

space. In most practical applications, an immense amount of computational resource and capital is needed to obtain the required dataset to reconstruct the model [7]. As a result, transfer learning is used. It can reduce the need for gathering resources and capital and at the same time use the pretrained model from a different application to work on the current application. Transfer learning uses learning from a preceding training model. Features learned by a CNN model originally trained on a huge dataset can be used to perform recognition tasks in a particular domain of the dataset [8–10]. Several methodologies for transfer learning have produced good results and there is a large literature on transfer learning and CNN. For example, Devikar [11] adopted transfer learning on 11 types of dog breeds, with 25 images for each class. He fine-tuned after retraining Google’s Inception V3 model from the dog dataset and achieved 96% accurate identification of the dog breed. The utility of the approach has also been demonstrated for plant images [12, 13], medical datasets [14–16], and a State Farm dataset [17] with an accuracy of up to 98% in the case of the Tapas [13]

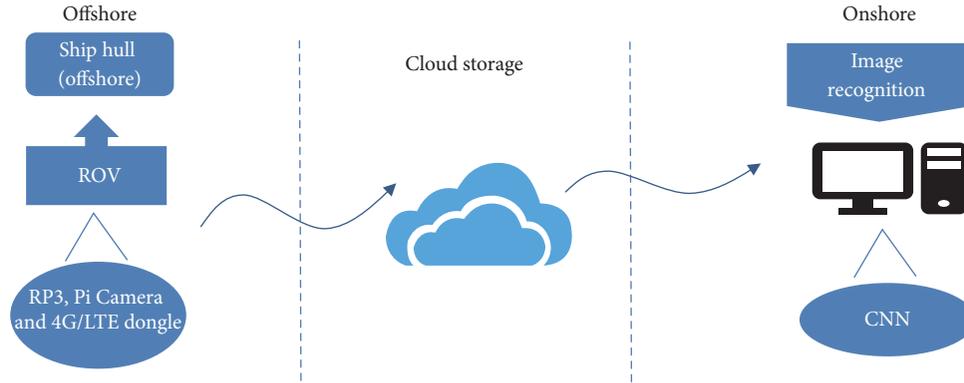


FIGURE 1: Proposed fouling recognition system design in this study.

study. In all these cases, a large database of good quality images with strong and distinctive features was required to produce good classification results. The transfer learning removes the need of a Graphics Processing Unit (GPU) for training despite the advantage of shortening the training time. The transfer learning surpasses full training of a CNN model on classification accuracy. It also reduces the need to label data by using features already learned from the previous model.

Although this approach has also been applied to water filtration and thermal investigations of fouling, for example, to predict fouling in the filtration of drinking water [18], heat exchangers [19], and heaters [20], transfer learning and deep CNN have not been applied to biofouling of marine structures such as ships' hulls. Notably, the above-mentioned classification results yielded more than 70% accuracy after using the transfer learning approach on CNN. This value is used as a benchmark for the present study, which is focussed on deploying transfer learning using Google's Inception V3 model [21, 22] with Softmax on a fouling database with ten categories and 1825 images. Retraining of the fouling database was performed using TensorFlow [23, 24] Docker image deep learning library.

In summary, the application of the Softmax transfer learning and deep convolutional neural networks on marine fouling recognition has not been reported in the literature. The early detection of marine growth through image recognition provides a means for the vessel's owner to schedule maintenance, for example, by hull grooming [25, 26], before the marine growth seriously compromises hull performance. Hence, the contributions made by the paper are as follows. The images of fouling taken from the ship can be transferred remotely to onshore for fouling recognition. The transfer learning and deep convolutional neural network can be used to classify the types and density of fouling using the captured image for early detection of fouling.

The remaining sections are organized as follows. Section 2 describes the fouling image recognition system followed by the proposed deep convolutional neural network in Section 3. Lastly, Section 4 concludes the paper.

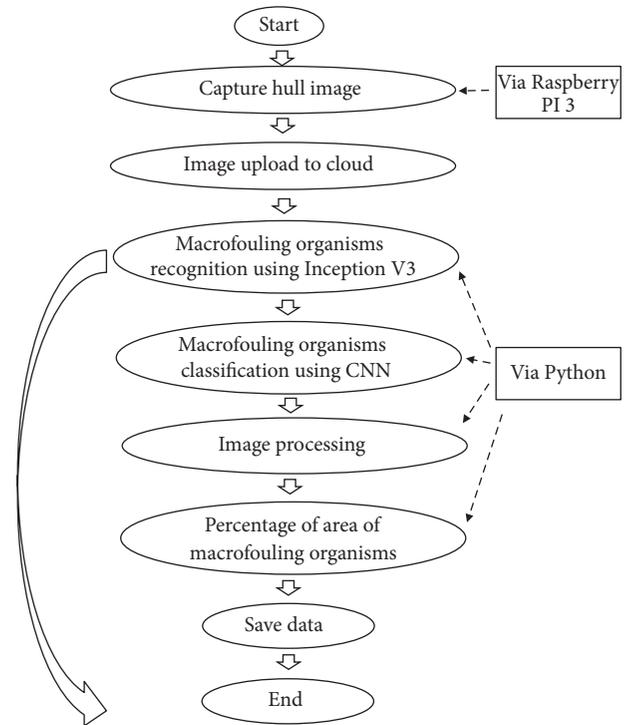


FIGURE 2: Software program flowchart.

2. Fouling Image Recognition System

The project aims to take a picture of a particular area on the surface of a ship hull via the camera module in the microcontroller. Due to field testing constraints, the captured images were obtained from the web. Some of the images do not truly represent the actual fouling on ships' hulls. The captured image will be stored in cloud storage via 4G/Long-Term Evolution (LTE) dongle. It will be automatically uploaded to the Cloud in Figure 1. The image will be used for fouling analysis via Deep Learning and CNN. As shown in Figure 2, the first stage of the program will identify if there are any macrofouling organisms on the



FIGURE 3: Different classes of fouling images used for training.

TABLE 1: Different classes and number of fouling images.

Fouling image class	Image count
Rock oysters	82
Kelp	272
Zebra mussel	220
Hydrozoan	116
Acorn barnacle	83
Algae	195
Finger sponge	119
Gooseneck barnacle	288
Christmas tree worm	222
Tunicates	228

surface of the ship hull using camera in Raspberry Pi. It is performed by carrying out image recognition using the Inception V3 model that was trained for ImageNet Large Scale Visual Recognition Challenge [21]. If the result indicates that there are no macrofouling organisms, the algorithm will stop. The percentage of macrofouling on the surface or fouling density will be determined. Otherwise, it will proceed to the next stage where the macrofouling organisms will be classified according to different classes of marine fouling as seen in Table 1 and Figure 3. There are a total of 1825 images in the database. The classes of marine fouling are

not exhaustive as there are more than 4,000 known fouling species. The fouling in Table 1 was decided by adapting the few common types of fouling on the vessels which allows the model to learn sufficient features for the classifier to perform its task. Due to the inconsistency in image size, quality, and differences in the fouling images, numbers of fouling images for each class are unequal. The images will be refined to be more representative of marine biofouling on ships' hulls and at earlier stages of development. In the last stage, the algorithm will detect the percentage of macrofouling organisms over the total area of the image known as the fouling density. This is achieved by processing the image via Open Source Computer Vision Library (OpenCV) using the color-based segmentation method. The various results obtained throughout the program can be saved. A Graphic User Interface (GUI) was designed to facilitate the process of running the different stages of the algorithms.

The arrangement of the image recognition system for biofouling is shown in Figure 4. It was not possible to arrange the test on board a ship. Instead, a laboratory setup was used. The fouling images were secured to a wall to simulate the presence of the fouling on the ship's hull. The left-hand side shows the microprocessor connected via mobile connection (situated on board the "ship") into cloud storage and to the local machine on the right side. The host device ("onshore") is used for fouling recognition and analysis. The microprocessor (i.e., Raspberry Pi 3 Model B) operates on Raspbian OS (see Figure 5), and the local machine uses

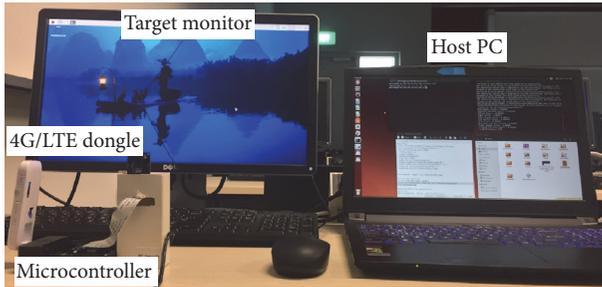


FIGURE 4: Overview of fouling image recognition system setup includes microprocessor connected via mobile connection (left) and host device (right) in laboratory.

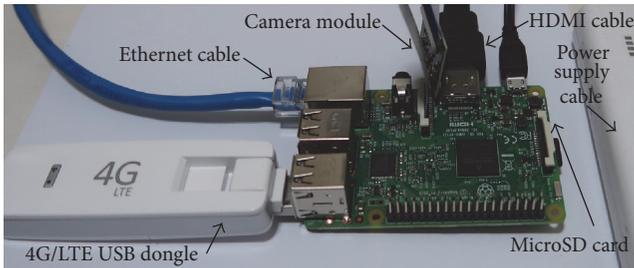


FIGURE 5: Microcontroller connection for fouling image recognition system.

Ubuntu OS. It can function as a computer with a built-in camera for building a smart device like fouling recognition system. For completeness, Tables 2 and 3 show the hardware and software components in the proposed system.

3. Proposed Deep Convolutional Neural Network

Convolutional neural network (CNN) is a type of deep learning neural network (NN) for image recognition and classification. This kind of neural network is made out of layers linked by artificial neurons that form a connection between the layers. The link carries valued weights that are fine-tuned throughout the training process, resulting in a trained network. The layers are constructed such that the first layer identifies a set of simple patterns of the input. A standard CNN contains 5 to 25 such layers and ends with an output layer [27]. Figure 6 shows a typical architecture of CNN used in this paper.

In the first convolution layer, features of inputs are extracted, such as edges, lines, and corners of the image. Taking a 3×3 feature extractor in a $n \times n$ size image, as an example and starting from the top left corner of an image, the feature extractor performs matrix multiplication and addition with the pixel value and then sums them. The summed value will enter the first hidden layer, which is also known as the first feature map. The feature extractor now slides itself by stride equal to 1 to the right and repeats the process. When the feature extractor reaches the far right of

TABLE 2: Hardware components.

Hardware name	Quantity	Remarks
Raspberry Pi 3 Model B	1	Microcontroller
Raspberry Pi camera	1	Image capturing
Raspberry Pi micro USB power supply	1	Power supply
HUAWEI 4G dongle	1	Cellular network dongle
4G LTE SIM card	1	Cellular network provider

TABLE 3: Software components.

Software name	Quantity	Purpose
Raspbian OS	1	OS for RP3
Ubuntu OS 14.04	1	OS for local machine
Python 3	1	Programming tool
TensorFlow Docker image	1	Deep learning library
Spider	1	Code editing tool

the image, it will move vertically down by stride equal to 1, return to the far left of the image, and repeat the process again. This process will continue until the 3×3 feature extractor reaches the bottom right of the image [27].

The pooling layer follows the convolutional layers. Pooling layers build up resistance against noise and distortion in the features. Maximum pooling and average pooling are the two ways to perform this task. Taking the example of a 4×4 feature map input into the pooling layer, for 2×2 pooling, the 4×4 input will be split into four matrices that do not lap over one another. In max pooling, the 2×2 pooling will only consider the largest value within itself to be the output. For average pooling, the 2×2 pooling will consider the average of the four values within itself to be the output. The advantage of pooling in image recognition is that it is invariant to slight shifting and rotating of the image. Therefore, pooling is a process that condenses the convolutional layers [27].

The nonlinear layer is an additional function that is applied after every convolution. The CNN takes advantage of this function by suggesting the prominent classification of likely similar features within the hidden layers. The CNN also uses various nonlinear functions and the most popular function is the rectified linear units (ReLU). The ReLU operates on a function of each pixel and substitutes all negative pixel values in the feature map by 0. The ReLU is needed as convolution is a linear operation and the data introduced into CNN is always nonlinear [27].

The fully connected layer is used as the last layer of CNN. The fully connected layer performs a summation of weights of features from its preceding layers. It implies that high-level features of outputs obtained from previous convolution and pooling are being classified based on the input image and the dataset the CNN is trained on. This classification will be brought over to the output layer where the results will be shown as the probability of result or class probability as seen in Tables 4 and 5.

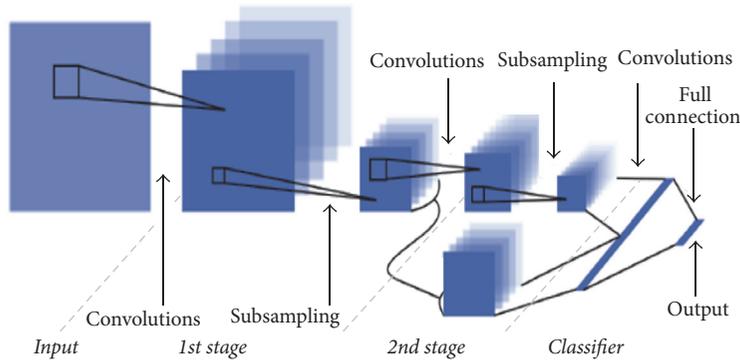


FIGURE 6: Architecture of convolutional neural network [10].

TABLE 4: Classification results of fouling (first and second training).

Classes	First training (%)	Second training (%)	Percent of improvement (%)
Rock oysters	75.945	96.236	20.291
Kelp	83.817	99.490	15.673
Zebra mussel	77.143	98.350	21.207
Hydrozoan	58.841	92.032	33.191
Acorn barnacle	62.001	96.771	34.770
Algae	58.749	89.231	30.482
Finger sponge	39.369	80.767	41.398
Gooseneck barnacle	89.061	99.903	10.842
Christmas tree worm	65.098	95.908	30.810
Tunicates	86.042	96.344	10.302

TABLE 5: Final testing results of fouling classification.

Classes of fouling	Test results (%)
Rock oysters	99.703
Kelp	99.128
Zebra mussel	87.916
Hydrozoan	97.635
Acorn barnacle	77.228
Algae	79.255
Finger sponge	76.617
Gooseneck barnacle	91.430
Christmas tree worm	94.961
Tunicates	99.171

The proposed Inception V3 model comprises three convolutional layers with a pooling layer, the three convolutional layers, ten inception modules, and a fully connected layer. There are a total of 17 layers in this model that contains features learned from the original training with ImageNet. The original fully connected layer will be retrained by a new layer. The activators of the bottleneck layer are used to produce bottleneck values that are placed in the Softmax

classifier. The new Softmax function will map the input image data to obtain the classification results [28].

3.1. Transfer Learning Using Inception V3 Model with Softmax.

The transfer learning that utilizes a pretrained neural network was implemented to identify and recognize the macrofouling organisms during the first stage of the program. This section shows the result of transfer learning using Inception V3 model with Softmax on the fouling image. Instead of training a deep network from scratch, a network trained on a different application is used. In this project, an image recognition model known as Inception V3 [21] was chosen. It consists of two main parts, namely, the feature extraction with a CNN and the classification part with fully connected and Softmax layers. The Inception V3 is capable of achieving good accuracy for image recognition of macrofouling organisms. Before the retraining process, the fouling database is arranged in the labeled directories. The ten classes of the dataset are stored in the directory of "fouling photos." The retraining process in TensorFlow Docker image was executed. The TensorFlow Docker helps to ease the starting and running of the TensorFlow open source software library. The first retraining process runs on 500 training steps with default train batch size of 10 and a learning rate of 0.01. The bottleneck values of each fouling image are stored in the bottleneck

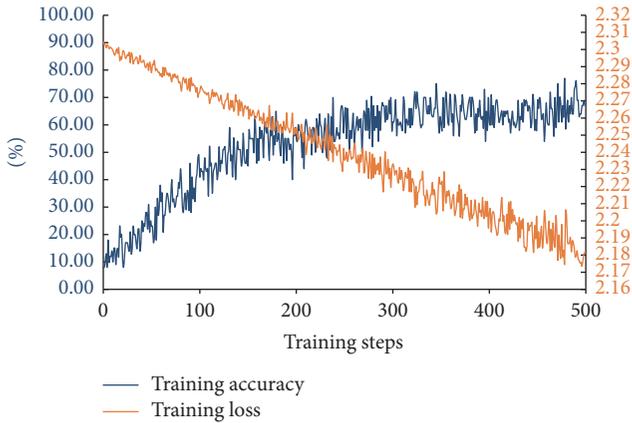


FIGURE 7: Training accuracy and loss for 1st training.

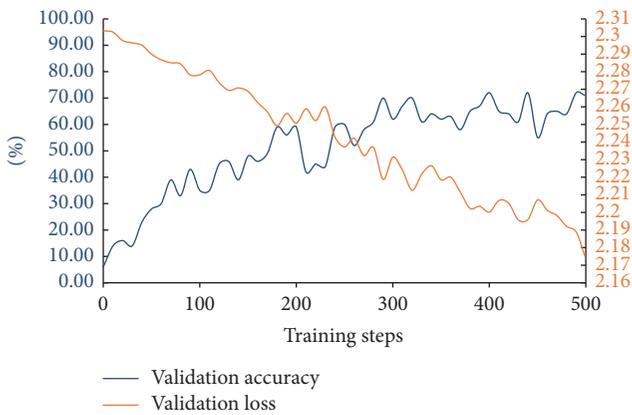


FIGURE 8: Validation accuracy and loss for 1st training.

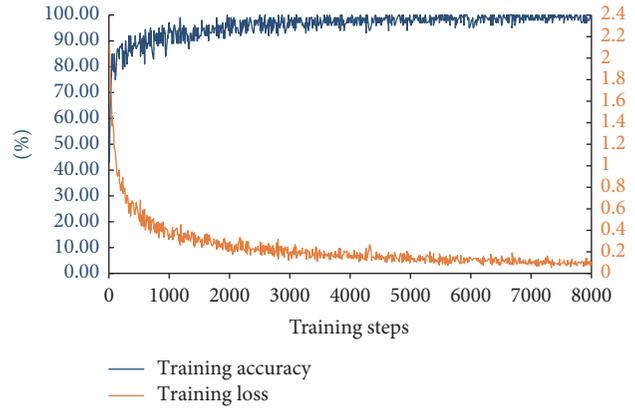


FIGURE 9: Training accuracy and loss for 2nd training.

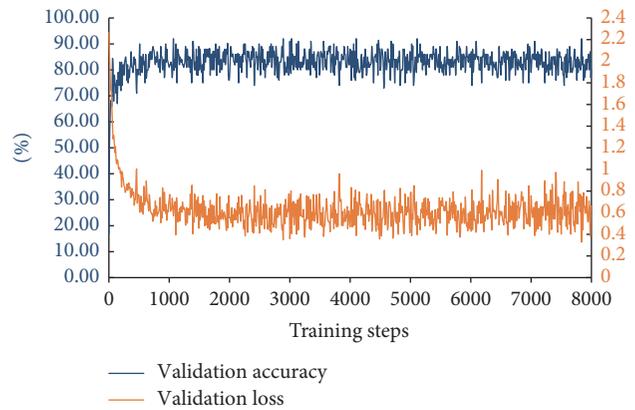


FIGURE 10: Validation accuracy and loss for 2nd training.

directory. The predictions are compared with the actual labeled image. The evaluation process will update the final layer's weights through a backpropagation process.

The training accuracy indicates the probability that the fouling images were identified correctly. The validation accuracy equates to the probability that any chosen image was identified correctly. The cross entropy is a function that reflects how far apart the images being classified are from their ground truth labels. A new retrained Inception V3 model graph, retrained labels text file containing ten fouling classes, and retrained log files are produced after the retraining process. The log files map out the training and validation accuracies with their losses. The first retraining process is plotted as shown in Figures 7 and 8. Random fouling photos were chosen from the dataset to test the new Softmax classifier. With the Softmax function, the input image was classified and sorted in order of confidence. As shown in Table 4, some of the classes such as hydrozoan, acorn barnacle, algae, finger sponge, and Christmas tree worm produced a result of less than 70%, which as mentioned above was set as the benchmark for what is acceptable.

Hence, the fouling samples in the fouling database are retrained to improve the validation accuracy. The retraining

process repeats until a higher final test accuracy is obtained. A set of random fouling images were used to test the classifier. The results are tabulated as shown in Table 4. Further retraining iterations were performed with different batch size and learning rate. However, it did not produce a better result. Moreover, the final test accuracy is worse than the first training. It was found that the fouling photos of algae did not possess sharp features. More images of algae were then downloaded for retraining. Some amendments were performed to produce an image of better quality with less noise. The second training was then repeated with the initial batch size and learning rate. Table 4 shows around 10 to 40% improvement in the second training. The final test in Table 5 shows improvement in the classification. For example, Figures 9 and 10 show the results of the second training for algae. The average result improved to 79.255%. Most of the previous methodologies of transfer learning generated classification accuracy of over 70%. In this paper, the classification results produce satisfactory classification results above 70% as shown in Table 5.

3.2. Macrofouling Recognition. The image processing algorithm was then tested on acorn barnacles that are a common type of macrofouling organism. This task of detecting the

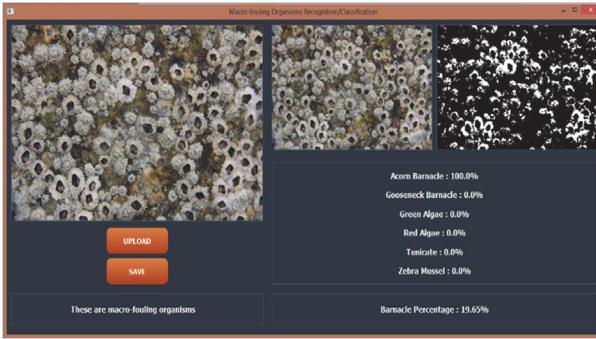


FIGURE 11: Proposed graphical user interface for image fouling recognition.

presence of macrofouling was accomplished via OpenCV. However, the algorithm can be modified to consider another type of fouling. Both the Red; Green; Blue (RGB) and Hue; Saturation; Value (HSV) color spaces have been tested. The RGB color model is more efficient and accurate than HSV. Fouling species of barnacle, especially in the tropics, are commonly red or white with red or purple stripes. But the shells may be colonised or overgrown by other fouling forms, which will complicate recognition/classification. Shapes of the openings of the shells will also vary depending on species besides the color. For a start, the color-based segmentation method was then implemented, and the morphological (shape) operation was carried out to process the image. An RGB lower and upper limit were decided for an off-white color similar to the color of acorn barnacles in the images used. A mask was then used to obtain the desired barnacle color. The image was then converted into greyscale. The morphological closing operation was used to close small holes inside the foreground objects. The number of 1 (i.e., white) that represents the acorn barnacle pixels was determined as shown in Figure 11. The total area of the image was also computed. The number of acorn barnacle pixels was divided over the entire area of the image to obtain the percent of acorn barnacle fouling. The routine used in this project relies on the CNN, and the accuracy depends on the size of the dataset for training.

A GUI is developed to allow users to interact with the functions as seen in Figure 11. The GUI was developed using PyQt5 to facilitate the flow of the algorithm, namely, recognition of macrofouling organisms by OpenCV, classification of macrofouling organisms via CNN model, and lastly the percent of acorn barnacle fouling. The GUI allows the users to upload the image via the “UPLOAD” button. The image and results of the classification and fouling density will be displayed as shown in Figure 12. If the image uploaded is nonmacrofouling organisms, the function buttons will be disabled. The results can be saved via the “SAVE” button to allow users to save the results for further analysis. Around 60 positive images (10 from each class shown in Figure 11) and 60 negative images (any other images) with size 720×480 in dimension were obtained from the Internet to test the accuracy of the recognition. Of the 60 positive images

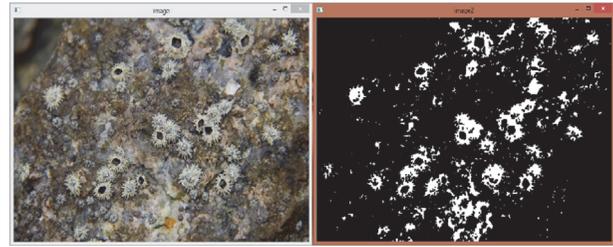


FIGURE 12: Acorn barnacle percent fouling over the total area.

tested, 53 images were correctly identified as macrofouling organisms while seven images were wrongly identified as nonmacrofouling organisms, giving an accuracy of 88%. Of the 60 negative images tested, 51 images were correctly identified as nonmacrofouling organisms while nine images were wrongly identified as macrofouling organisms, giving an accuracy rate of 85%. The mean accuracy for this experiment is, therefore, 86.50%, which is relatively high given that a pretrained model was used. The results show that the image processing algorithm can detect the presence of macrofouling organisms.

3.3. Macrofouling Classification by CNN. Further simulations were carried out on two different datasets to determine whether the accuracy will be affected by the different number of images for each class. In this study, the quality and relevance of the images are acceptable for training. Dataset #1 contained a total of 1423 images, with 997 and 426 containing the training and validating images, respectively. Dataset #2 contained a total of 582 images, with 402 and 180 being training and validating images, respectively. For both datasets, epochs (or a number of steps) were set at 100/75/50/25, respectively. As seen from Table 6, the accuracy increases with the number of epochs and the size of the dataset. A larger number of epochs and datasets should be used to train the CNN model to improve the training accuracy. A larger network with a greater number of layers can be implemented to improve the accuracy. However, the extensive neural network will cause severe overfitting and will be computationally expensive to train.

After training the CNN models, approximately 60 positive images used in previous training were used to test the different trained models to compare their accuracies in classification. As observed in Table 7, the accuracy increases with the number of epochs. The classification of the macrofouling organisms gave a mean accuracy of 74.75%, median of 70.50%, and standard deviation of 7.92%.

3.4. Macrofouling Density. The acorn barnacle images were used to obtain the percent fouling over the total area of the image or the fouling density by OpenCV. Around 40 acorn barnacle images were used for the test. The results obtained gave the lowest fouling density of 8.58%, highest value of 36.79%, and mean of 20.18%. Based on the results obtained, the fouling density will be categorized. For example, a range of 5%–15% will be considered light fouling, 15%–25%

TABLE 6: CNN models training results.

Data	Number of images	Training images	Validating images	Model name	Epochs	Batch size	Loss	Accuracy	Val. loss	Val_acc
Dataset 1	1423	997	426	CNN_Model.h1	100	96	0.0166	0.9354	0.0803	0.6879
				CNN_Model.h2	75		0.0278	0.8864	0.0743	0.6848
				CNN_Model.h3	50		0.0475	0.8062	0.0902	0.6303
				CNN_Model.h4	25		0.0669	0.7206	0.0673	0.7091
Dataset 2	582	402	180	CNN_Model.h5	100	96	0.0191	0.9225	0.1046	0.6250
				CNN_Model.h6	75		0.0242	0.9182	0.1041	0.6250
				CNN_Model.h7	50		0.0302	0.8916	0.0670	0.6771
				CNN_Model.h8	25		0.0617	0.7351	0.0836	0.6146

TABLE 7: CNN models test results.

Data	Model name	Epochs	Batch size	Acorn barnacles	Gooseneck barnacle	Algae	Tunicate	Zebra mussel	Accuracy
Dataset 1	CNN_Model.h1	100	96	6/10	10/10	10/10	9/10	9/10	89%
	CNN_Model.h2	75		8/10	10/10	9/10	9/10	6/10	87%
	CNN_Model.h3	50		10/10	6/10	8/10	7/10	6/10	69%
	CNN_Model.h4	25		8/10	7/10	8/10	9/10	1/10	67%
Dataset 2	CNN_Model.h5	100	96	4/10	8/10	7/10	10/10	4/10	72%
	CNN_Model.h6	75		8/10	6/10	8/10	10/10	3/10	69%
	CNN_Model.h7	50		9/10	6/10	8/10	9/10	6/10	78%
	CNN_Model.h8	25		9/10	2/10	9/10	9/10	3/10	67%

medium fouling, and 25%–40% heavy fouling. The routine is capable of determining the percentage of fouling and the level of fouling as shown in Figure 12. The test for the acorn barnacle can be modified to account for the different classes by defining the RGB color range based on their color feature. However, there are some limitations for the tunicate class as they exist in many different colors and morphologies. In summary, the transfer learning using Google’s Inception V3 model with Softmax at last layer was successfully performed on a fouling database of 10 categories and 1825 images. The fouling was classified correctly with over 70% validation accuracy. The image processing approach can also detect the percent of fouling on a given surface area of interest.

4. Conclusion

Marine biofouling has adverse effects on marine vessel operational performance and costs. The rapid and low-cost implementation of image recognition through classification via transfer learning on a pretrained convolutional neural networks (CNN) model (named Inception V3) provides a potential solution for early detection of fouling on ship’s hull. The wireless transmission via mobile network enables a fouling image to be uploaded to Google Drive cloud storage easily and subsequently used for image recognition. The graphical user interface was developed with the functions to facilitate the flow of the fouling recognition algorithms from fouling recognition and fouling classification to the density of fouling via CNN and Open Source Computer Vision Library (OpenCV). The acorn barnacle was recognized and

classified with acceptable validation accuracy. The percent of fouling due to acorn barnacle on given surface area was also determined successfully.

This study has the desired benchmark in the final fouling classification accuracy and fouling density. Future work will develop a database of “real-world” images of a greater diversity of fouling organisms at different stages of development and examine the potential of the CNN approach for remote assessment of fouling. Other types of unsupervised machine learning techniques will be used to compare with the proposed CNN approach. The shapes of the openings of the species will be included in macrofouling recognition. The testing of the image recognition system will be conducted on board the ship via a maritime satellite broadband system such as VSAT or SEVSAT.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to express the deepest appreciation to Mr. Goh Jun Yi, Mr. Galvan Goh Jia Xuan, and Mr. Clauson Seah Jie Tai who graduated from Newcastle University for sharing their reports on fouling recognition and classification. The authors would like to acknowledge the Global Excellence Fund Award (ID: ISF#51) from Newcastle University for sponsoring and supporting the project since 2015.

References

- [1] J. A. Lewis, "Marine biofouling and its prevention on underwater surfaces," *Materials Forum*, vol. 22, pp. 41–61, 1998.
- [2] A. S. Clare, D. Rittschof, D. J. Gerhart, and J. S. Maki, "Molecular approaches to nontoxic antifouling," *Invertebrate Reproduction and Development*, vol. 22, no. 1-3, pp. 67–76, 1992.
- [3] M. Wahl, *Marine Epibiosis. 1. Fouling and Antifouling - Some Basic Aspects*, vol. 58, Marine Ecology Progress Series, 1989.
- [4] M. P. Schultz, "Effects of coating roughness and biofouling on ship resistance and powering," *Biofouling*, vol. 23, no. 5, pp. 331–341, 2007.
- [5] M. P. Schultz, J. A. Bendick, E. R. Holm, and W. M. Hertel, "Economic impact of biofouling on a naval surface ship," *Biofouling*, vol. 27, no. 1, pp. 87–98, 2011.
- [6] F. T. Chan, H. J. Macisaac, and S. A. Bailey, "Relative importance of vessel hull fouling and ballast water as transport vectors of nonindigenous species to the canadian arctic," *Canadian Journal of Fisheries and Aquatic Sciences*, vol. 72, no. 8, pp. 1230–1242, 2015.
- [7] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [8] J. Donahue, Y. Jia, O. Vinyals et al., "DeCAF, a deep convolutional activation feature for generic visual recognition," *In Icml*, vol. 32, pp. 647–655, 2014.
- [9] G. Levi and T. Hassner, "Age and gender classification using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPRW 2015*, pp. 34–42, usa, June 2015.
- [10] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *Proceedings of the International Joint Conference on Neural Network (IJCNN '11)*, pp. 2809–2813, IEEE, August 2011.
- [11] P. Devikar, "Transfer Learning for Image Classification of various dog breeds," *International Journal of Advanced Research in Computer Engineering and Technology (IJARCET)*, vol. 5, no. 12, pp. 2707–2715.
- [12] A. K. Reyes, J. C. Caicedo, and J. E. Camargo, "Fine-Tuning Deep Convolutional Networks for Plant Recognition," *fra*, September 2015, 1391, 1–9.
- [13] A. Tapas, "Transfer learning for image classification and plant phenotyping," *International Journal of Advanced Research in Computer Engineering and Technology (IJARCET)*, vol. 5, no. 11, pp. 2664–2669.
- [14] H. Shin, H. R. Roth, M. Gao et al., "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1285–1298, 2016.
- [15] N. Bayramoglu and J. Heikkilä, "Transfer Learning for Cell Nuclei Classification in Histopathology Images," in *Computer Vision—ECCV 2016 Workshops*, pp. 532–539, 2016.
- [16] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-scale Image Recognition." arXiv preprint arXiv:1409.1556, 2014.
- [17] D. Singh, "Using Convolutional Neural Networks to Perform Classification on State Farm Insurance Driver Images," 4321–4325.
- [18] N. V. Delgrange, C. Cabassud, M. Cabassud, D. L. Bourlier, and J. M. Lainé, "Neural networks for long term prediction of fouling and backwash efficiency in ultrafiltration for drinking water production," *Desalination*, vol. 131, no. 1-3, pp. 353–362, 2000.
- [19] S. Lalot and H. Pálsson, "Detection of fouling in a cross-flow heat exchanger using a neural network based technique," *International Journal of Thermal Sciences*, vol. 49, no. 4, pp. 675–679, 2010.
- [20] S. Lalot and S. Lecoeuche, "Online fouling detection in electrical circulation heaters using neural networks," *International Journal of Heat and Mass Transfer*, vol. 46, no. 13, pp. 2445–2457, 2003.
- [21] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS '12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.
- [22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR '16)*, pp. 2818–2826, July 2016.
- [23] M. Abadi, P. Barham, J. Chen et al., "TensorFlow, A system for large-scale machine learning," in *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI, Savannah, Ga, USA, 2016)*.
- [24] B. J. Erickson, P. Korfiatis, Z. Akkus, T. Kline, and K. Philbrick, "Toolkits and Libraries for Deep Learning," *Journal of Digital Imaging*, pp. 1–6, 2017.
- [25] M. Tribou and G. Swain, "The use of proactive in-water grooming to improve the performance of ship hull antifouling coatings," *Biofouling*, vol. 26, no. 1, pp. 47–56, 2010.
- [26] M. Tribou and G. Swain, "Grooming using rotating brushes as a proactive method to control ship hull fouling," *Biofouling*, vol. 31, no. 4, pp. 309–319, 2015.
- [27] S. Hijazi, R. Kumar, C. Rowen, Group IP, and Cadence, "Using Convolutional Neural Networks for Image Recognition," http://owl.english.purdue.edu/owl/resource/560/01/https://ip.cadence.com/uploads/901/cnn_wp-pdf.
- [28] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwicki, "Transforming sensor data to the image domain for deep learning — An application to footstep detection," in *Proceedings of the 2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2665–2672, Anchorage, AK, USA, May 2017.

Research Article

Adaptive Neural Network Sliding Mode Control for Quad Tilt Rotor Aircraft

Yanchao Yin, Hongwei Niu, and Xiaobao Liu

Faculty of Mechanical and Electrical Engineering, Kunming University of Science and Technology, Kunming, Yunnan 650500, China

Correspondence should be addressed to Yanchao Yin; yinyc@163.com

Received 26 June 2017; Accepted 21 August 2017; Published 11 October 2017

Academic Editor: Guang Li

Copyright © 2017 Yanchao Yin et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A novel neural network sliding mode control based on multicommunity bidirectional drive collaborative search algorithm (M-CBDCS) is proposed to design a flight controller for performing the attitude tracking control of a quad tilt rotors aircraft (QTRA). Firstly, the attitude dynamic model of the QTRA concerning propeller tension, channel arm, and moment of inertia is formulated, and the equivalent sliding mode control law is stated. Secondly, an adaptive control algorithm is presented to eliminate the approximation error, where a radial basis function (RBF) neural network is used to online regulate the equivalent sliding mode control law, and the novel M-CBDCS algorithm is developed to uniformly update the unknown neural network weights and essential model parameters adaptively. The nonlinear approximation error is obtained and serves as a novel leakage term in the adaptations to guarantee the sliding surface convergence and eliminate the chattering phenomenon, which benefit the overall attitude control performance for QTRA. Finally, the appropriate comparisons among the novel adaptive neural network sliding mode control, the classical neural network sliding mode control, and the dynamic inverse PID control are examined, and comparative simulations are included to verify the efficacy of the proposed control method.

1. Introduction

It is well known that quad tilt rotors aircraft (QTRA) can be taken as a specific robotic system, whose dynamic modeling and control can be further tailored for further application to generic robotics. In particular, QTRA has been receiving an increasing attention among the global researchers, due to its specific capabilities combined with the advantages of both fixed-wing aerial and helicopters, for example, high cruising speed, long flight range, large loading, and ability of vertical take-off and landing (VTOL) in limited space [1]. Hence, they are more suitable for commercial, firefighting, and investigation purposes, such as assembly of large pieces, natural disasters assessment, reconnaissance of forest fires, and traffic surveillance [2]. However, as a specific robotic manipulators, QTRA is an underactuated system with six-freedom degree and 4 inputs, which rely on wings and rotors for generating lift; that is, it will have to deal with serious aerodynamic interference between rotors and wings due to their positional relationship [3]. Consequently, the

mechanical structure of the QTRA and its strong nonlinearities, unknown external disturbances, coupled dynamics, and multivariable properties, increase the complexity of attitude control designs. In fact, the control design for QTRA has been relatively immature, and any inappropriate control strategy could lead to instability issues, such as the 4 major accidents in the tilt rotor V-22 caused by out of attitude control in the VTOL stage.

With respect to this challenging issue, a considerable amount of effort has been invested for the attitude control of QTRA. In particular, the PID and LQR control methods have been well recognized, where nonlinear structure algorithms are used to suppress integral saturation [4, 5], and two linear structure algorithms have been more widely adopted to control integral saturation, including conditional integration [6] and feedback inhibition method [7]. In the subsequent studies, several modifications have been made on the integral part to improve the control capability against saturation, for example, integral part prediction PID modification [8], variable structure PID modification [9], and multimode PID

modification [10], where the value of integral part will eventually be adjusted with the saturation of controller. However, the majority of these available PID control methods all impose an assumption that the disturbances to be considered are none or varying lowly.

In practical flight scenarios, the uncertainties in the system parameters along the attitude varying may be unavoidable due to complex couplings between the attitude and position, the unknown external disturbances, or the modeling uncertainties and attitude constraints that affect the control system dynamics [11, 12]. Consequently, the precise attitude control is of great importance for the control system design of QTRA. To deal with this issue, some efforts have been advanced to exploit the ability of robustness, invariance to uncertainties, and resistance to external disturbance for sliding mode control (SMC) [13] to improve the robustness of flying craft control. In recent years, many researchers have investigated the incorporation of SMC into adaptive control and intelligent control methods, such as combining SMC with neural networks [14] or with fuzzy logic systems [15]. Particularly, in the application of SMC, its combination with neural network is validated to overcome an obvious shortcoming; that is, the chattering problem that impedes the application of SMC.

It is well known that radial basis function (RBF) neural network can locally approximate unknown dynamic functions with arbitrary precision in certain conditions, which has strong self-learning, adaptation, and fault-tolerant capability [16, 17]. Accordingly, RBF neural network has been adopted for SMC systems to eliminate chattering phenomenon. It is shown in [18] that the sliding mode variable structure controller using a RBF neural network is more robust to disturbances with a state observer. In [19, 20], the application of neural network to estimate the changeable control law and combined boundary layer integral to eliminate the chattering are validated to improve the dynamics of system. It is noted that the presented methods depend on the integral boundary expansion or observer design, where the induced expansion-boundary or observer errors are used to reduce the switching frequency of control law for improving the robustness of the traditional sliding mode controller. Moreover, the disturbances and the potential approximation errors are not considered in [21]. Following this framework, the authors of [22] proposed a neural network adaptive backstepping high-order terminal sliding mode control scheme, where neural networks are employed to approximate the unknown nonlinear functions, and a high-order sliding mode control law using the nonsingular terminal is designed to eliminate chattering. Although it was proved that the system is robust to both matched and mismatched uncertainties, the control law of lower-order emerges order coupling during deriving the equivalent control law; that is, the controller is difficult to implement. Besides the observer design and control law regulation, fractional-order calculus theory [23] and discrete time terminal method [24] combining RBF neural network were also exploited to eliminate the chattering of SMC.

In particular, a complementary sliding mode controller is designed by adopting the combination of the generalized sliding surface and the complementary sliding surface to

improve the system dynamic performance and robustness [25]. In the latest work, a bioinspired algorithm has attracted researchers to apply such intelligent approaches to optimize the network structure parameters, and the sliding mode error is introduced in the adaptive law to improve the performance of the systems by regulating the centers, widths, and weight of RBF network online. Noticeably, for the attitude control of QTRA with unknown external disturbances and modeling errors, the long convergence time and significant chattering are still a problem to be solved in most of the aforementioned results.

According to the sliding mode control principle, if the system converged to the sliding surface, the performance of sliding mode can be greatly improved. Motivated by this fact and our recent work [26, 27], we proposed a novel adaptive neural network sliding mode control for QTRA attitude with external uncertain disturbances and multichannel couplings, where RBF neural network is used to online regulate the equivalent sliding mode control law, and a robust control law is added to eliminate the approximation error. Then an adaptive control algorithm is designed, where the attitude stability requirements concerning the propeller tension, channel arm, and moment of inertia are all studied. Especially, considering the influence of the hidden layer unit number and the network essential structure parameters on the validity of neural network approximation, a novel multicommunity bidirectional drive collaborative search algorithm is developed to uniformly update the unknown neural network weights and essential model parameters adaptively, where the nonlinear approximation error is obtained and serves as a novel leakage term in the adaptations. Consequently, the attitude of the control system is guaranteed to converge to the sliding surface stably, which can effectively improve the convergence rate and eliminate chattering phenomenon of equivalent sliding mode control law and benefit the overall attitude control performance for QTRA. Appropriate comparisons among the novel adaptive neural network sliding mode control, the classical neural network sliding mode control, and the dynamic inverse PID control are examined. Simulation results are provided to validate the efficacy of the proposed control method.

This paper is organized as follows: the problem to be studied and the attitude dynamics model of QTRA is stated in Section 2. Adaptive neural network sliding mode control with a novel multicommunity bidirectional drive collaborative search algorithm is presented in Section 3. Comparison simulation results are given in Section 4 and conclusion is made in Section 5.

2. Problem Formulation

In this section, the proposed QTRA in Figure 1 has 8 inputs of the motors, which includes 4 inputs of regular QTRA and the additional 1 input for each of 4 of the tilting motors, and they are not fixed and are able to tilt-roll separately from each other. In particular, the motors could rotate around pitch to compensate for unexpected effects occurred from internal or external environment. Nevertheless, the body fixed coordinate is defined, where the focal point O^b of

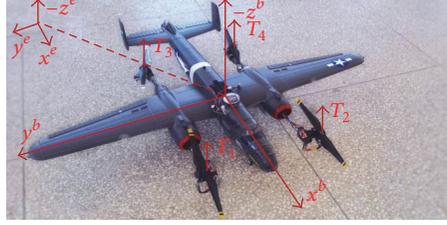


FIGURE 1: Coordinate systems: inertial earth fixed frame and body fixed frame.

gravity for QTRA upwards concurring is chosen as the Z^b axis, the perpendicular lever is the X^b axis, and the Y^b axis is defined by the right hand rule. Then X^b , Y^b , and Z^b axes are characterized individually as roll (ϕ), pitch (θ), and yaw (φ) angles, respectively. For inertial earth fixed frame, an east-north-up orthogonal coordinate system is created by setting the QTRA flight starting point O^e as the origin.

Consequently, a nonlinear attitude control system for QTRA aircraft with attitude angles defined as in Figure 2 is studied, where the QTRA is assumed to be a symmetric rigid body with constant mass, and the roll (ϕ) pitch (θ) and yaw (φ) angles have a little variation during the VTOL stage. The transformation matrix $R(\phi, \theta, \varphi)$ from inertial coordinates to body fixed coordinates is given by [23]

$$R(\phi, \theta, \varphi) = \begin{bmatrix} \cos(\varphi) \cos(\theta) & \cos(\varphi) \sin(\theta) \sin(\phi) - \sin(\varphi) \cos(\phi) & \cos(\varphi) \sin(\theta) \cos(\phi) + \sin(\varphi) \sin(\phi) \\ \sin(\varphi) \cos(\theta) & \sin(\varphi) \sin(\theta) \sin(\phi) - \cos(\varphi) \cos(\phi) & \sin(\varphi) \sin(\theta) \cos(\phi) - \sin(\varphi) \sin(\phi) \\ -\sin(\theta) & \cos(\theta) \sin(\phi) & \cos(\theta) \cos(\phi) \end{bmatrix}. \quad (1)$$

If the propeller pull and earth's gravitational force are represented as T and G , respectively, in Figure 2 and the resultant force of QTRA body frame is denoted by F_i , each of the axis thrust vectors \vec{F}_i can be given in (2) by using rotation matrix

$$\begin{aligned} \vec{F}_i &= \begin{bmatrix} F_{ix} \\ F_{iy} \\ F_{iz} \end{bmatrix} \cdot R(\phi_i, \theta_i, \varphi_i) \\ &= \begin{bmatrix} (T_1 + T_2 + T_3 + T_4) \sin \partial + mg \sin \theta \\ mg \sin \phi \cos \theta \\ (T_1 + T_2 + T_3 + T_4) \cos \partial - mg \cos \phi \cos \theta \end{bmatrix}, \end{aligned} \quad (2)$$

where F_i , F_{iy} , and F_{iz} are the thrust force of the components along the axis, respectively, T_1 , T_2 , T_3 , and T_4 are the pull for each propeller, and γ represents the rotor tilting angle.

Suppose L_x , L_y , and L_z as the arm length of the rotational torque for x -, y -, z -axis, respectively; each of the axis torques can be calculated as follows:

$$\begin{aligned} \vec{M}_i &= \begin{bmatrix} M_{ix} \\ M_{iy} \\ M_{iz} \end{bmatrix} \cdot R(\phi_i, \theta_i, \varphi_i) \\ &= \begin{bmatrix} (T_1 \cos \gamma + T_4 \cos \gamma) L_x - (T_2 \cos \gamma + T_3 \cos \gamma) L_x \\ (T_1 \cos \gamma + T_2 \cos \gamma) L_y - (T_3 \cos \gamma + T_4 \cos \gamma) L_y \\ (T_1 \sin \gamma + T_4 \sin \gamma) L_z - (T_2 \sin \gamma + T_3 \sin \gamma) L_z \end{bmatrix}. \end{aligned} \quad (3)$$

Therefore, the resultant torque of each axis can be calculated, respectively, as follows:

$$M_i = \sqrt{M_{ix}^2 + M_{iy}^2 + M_{iz}^2}. \quad (4)$$

According to the law of severity, the torque equation in (4) is produced by the rotation speeds of the propellers, such that (3) becomes

$$\sum M = \frac{\delta H}{\delta t} + \omega \times H, \quad (5)$$

where H is the moment of momentum of QTRA to the earth frame, and the component of rotation along the angular velocity vector can be given as follows:

$$\omega = pi + qj + rk. \quad (6)$$

If the moment of inertia to the body frame is denoted by I , the relation between angular momentum and angular velocity can be given as

$$H = I\omega. \quad (7)$$

The kinetic equation of centroids for QTRA can be derived in

$$\begin{aligned} M_x &= I_{xx} \dot{p} - I_{xy} \dot{q} - I_{xz} \dot{r} + (I_{zz} - I_{yy}) qr \\ &\quad + I_{yz} (r^2 - q^2) - I_{xz} p q + I_{xy} pr, \end{aligned}$$

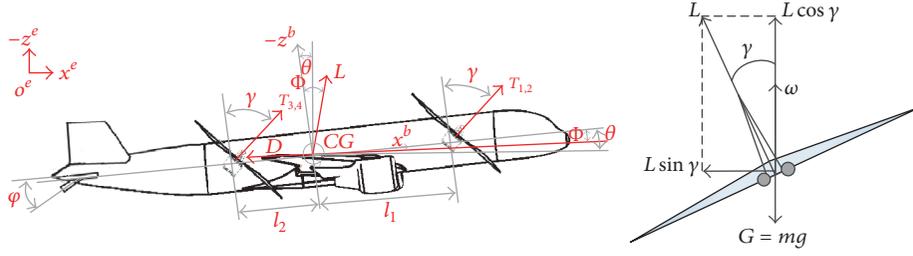


FIGURE 2: Body scheme showing the forces acting on the QTRA.

$$\begin{aligned}
 M_y &= -I_{xy}\dot{p} + I_{yy}\dot{q} - I_{yz}\dot{r} + (I_{xx} - I_{zz})pr \\
 &\quad + I_{xz}(p^2 - r^2) + I_{zy}pq - I_{xy}qr, \\
 M_x &= -I_{xz}\dot{p} - I_{zy}\dot{q} + I_{zz}\dot{r} + (I_{yy} - I_{xx})pq \\
 &\quad + I_{xy}(q^2 - p^2) + I_{xz}qr - I_{yz}pr,
 \end{aligned} \tag{8}$$

where I_{xx} , I_{yy} , and I_{zz} donate the rotary inertia of body frame axis and I_{xy} , I_{xz} , and I_{yz} are the inertial products, respectively.

In this study, the roll (ϕ) pitch (θ) and yaw (φ) are 1 to 5 degrees, respectively, by the user input during the VTOL stage. In this case we assume that (ϕ, θ, φ) equals (p, q, r) and $(\cos \phi, \cos \theta, \cos \varphi)$ equals $(1, 1, 1)$. Therefore, the attitude dynamics of QTRA can be derived with modeling errors and external disturbances using the formula in (9):

$$\ddot{\phi} = \frac{I_{xz}(I_{xx} - I_{yy} + I_{zz})\dot{\theta}\dot{\phi} + [I_{zz}(I_{yy} - I_{xx}) - I_{xz}^2]\dot{\phi}\dot{\phi} + I_{zz}L_\phi\partial_T(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) - I_{xz}L_\phi mg\gamma}{I_{xx}I_{zz} - I_{xz}^2} + \mathfrak{R}_1, \tag{9}$$

$$\ddot{\theta} = \frac{(I_{zz} - I_{xx})\dot{\phi}\dot{\phi} + I_{xz}(\dot{\phi}^2 - \dot{\phi}^2) + L_\theta\partial_T(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2)}{I_{yy}} + \mathfrak{R}_2, \tag{10}$$

$$\ddot{\varphi} = \frac{I_{xx}(I_{xx} - I_{yy})\dot{\theta}\dot{\phi} + I_{xz}(I_{xx} - I_{yy} + I_{zz})\dot{\phi}\dot{\phi} + I_{xz}L_\phi\partial_T(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2) - I_{xx}L_\phi mg\gamma}{I_{xx}I_{zz} - I_{xz}^2} + \mathfrak{R}_3. \tag{11}$$

In (11), \mathfrak{R}_1 , \mathfrak{R}_2 , and \mathfrak{R}_3 donate composite disturbance for each channel, and in this case we assume that $u_1 = L_\phi\partial_T(\omega_1^2 - \omega_2^2 + \omega_3^2 - \omega_4^2)$, $u_2 = L_\theta\partial_T(\omega_1^2 + \omega_2^2 - \omega_3^2 - \omega_4^2)$, and $u_3 = L_\phi mg\gamma$ stand for control inputs for each channel, respectively, where ∂_T donates tension coefficient of propeller, ω_i is the rotation velocity of propeller, m is the mass of aircraft, and L_ϕ , L_θ , and L_φ are the arm of each channel.

Consequently, the nonlinear system of attitude hold control with uncertainties can be written as (12) by defining $x = [x_1, x_2, x_3, x_4, x_5, x_6]^T$ as $\dot{h} = [\dot{\phi}, \dot{\phi}, \dot{\theta}, \dot{\theta}, \dot{\varphi}, \dot{\varphi}]$ using (9)–(11)

$$\begin{aligned}
 \dot{x}_{(2k-1)}(t) &= x_{(2k)}(t) \\
 \dot{x}_{(2k)}(t) &= f_k(x, t) + \Gamma_k u_k + \mathfrak{R}_k
 \end{aligned} \tag{12}$$

$k = 1, 2, 3,$

where $u_k = [u_1, u_2, u_3]^T$, $\mathfrak{R}_k = \Delta f_k(x, t) + \Delta \Gamma_k \ell_k + \Psi_k$, Δf and $\Delta \Gamma_k$ represent modeling errors, and Ψ_k represents external disturbance,

$$\Gamma_k = \begin{bmatrix} \frac{I_{zz}}{I_{xx}I_{zz} - I_{xz}^2} & 0 & \frac{I_{xz}}{I_{xx}I_{zz} - I_{xz}^2} \\ 0 & \frac{1}{I_{yy}} & 0 \\ \frac{I_{xz}}{I_{xx}I_{zz} - I_{xz}^2} & 0 & \frac{I_{xx}}{I_{xx}I_{zz} - I_{xz}^2} \end{bmatrix}. \tag{13}$$

3. Neural Network Sliding Mode Control Optimization Design

In this section, the design of an adaptive neural network sliding mode controller for QTRA is presented. First, the conventional equivalent sliding mode control for QTRA is designed. Then, the adaptive RBF neural network is proposed based on multicommunity bidirectional drive collaborative search algorithm (M-CBDCS), where the coefficient of switching control is adaptively calculated to overcome the chattering phenomenon of traditional equivalent sliding control. Finally, the stability of the adaptive neural network sliding mode control is proven using Lyapunov theory.

3.1. Equivalent Sliding Mode Control for QTRA. The sliding mode control law based on equivalent control usually consists of an equivalent control law u_{eqk} and a switching control law u_{swk} which can be obtained:

$$u_k = u_{eqk} + u_{swk}. \tag{14}$$

Conventionally, the equivalent law is deduced from the relationship between the sliding mode s and its differential \dot{s} on the basic of the pertinent mathematic mode of the system, which could ensure the state of system along the sliding surface. The switching control law is applied to compensate

for uncertainty and external disturbance, which ensures the state of system not leaving the sliding surface.

Considering the nonlinear system about QTRA, the tracking error between the desired reference and the system response is taken as

$$e_{(2k-1)} = x_{(2k-1)} - x_{d(2k-1)}, \quad (15)$$

where

$$\dot{e}_{(2k-1)} = \dot{e}_{(2k)}. \quad (16)$$

According to the theories of sliding mode control, the switching function for the sliding mode can be defined as follows:

$$s_k = c_k e_{(2k-1)} + \dot{e}_{(2k-1)}, \quad c_k > 0. \quad (17)$$

Combining (12)–(15) and (5), we can get

$$\begin{aligned} \dot{s}_k &= c_k \dot{e}_{(2k-1)} + \dot{e}_{(2k)} = c_k \dot{e}_{(2k-1)} + \dot{x}_{(2k)} - \dot{x}_{d(2k)} \\ &= c_k \dot{e}_{(2k-1)} - \dot{x}_{d(2k)} + f_k(x, t) + \Gamma_k u_k + \mathfrak{R}_k. \end{aligned} \quad (18)$$

According to $\dot{s}_k = 0$, we can get equivalent control of the sliding mode as follows

$$\begin{aligned} c_k \dot{e}_{(2k-1)} - \dot{x}_{d(2k)} + f_k(x, t) + \Gamma_k u_{\text{eq}_k} + \mathfrak{R}_k &= 0, \\ u_{\text{eq}_k} &= \frac{1}{\Gamma_k} (-c_k \dot{e}_{(2k-1)} + \dot{x}_{d(2k)} - f_k(x, t) - \mathfrak{R}_k). \end{aligned} \quad (19)$$

To ensure the sliding mode reaching condition $s(t) \cdot \dot{s}(t) \leq -\eta |s(t)|$, $\eta > 0$, the switching law can be designed as

$$u_{\text{sw}_k} = -\eta \cdot \frac{\text{sgn}(s_k)}{\Gamma_k}, \quad (20)$$

where

$$\text{sgn}(s_k) = \begin{cases} 1, & s_k > 0 \\ -1, & s_k < 0. \end{cases} \quad (21)$$

Remark 1. The switching law shown in (20) is mainly used to compensate for the uncertainties and external disturbances so that system state can retain on the sliding surface and move along it. However, the switching law is actually a nonlinear set of discontinuous functions leading to the chattering problem.

3.2. Adaptive Neural Network Sliding Mode Control. Usually, the equivalent law of sliding mode control depends heavily upon the models and the accuracy of parameters. However, in most cases, these models of complicated system always exist in simplified or linear form by regulating its parameters in different working conditions. Therefore, to compensate for the inaccuracy arising from these uncertainties, the switching law becomes necessary and critical.

In order to overcome the chattering phenomenon of traditional sliding mode control effectively, (20) is replaced by the adaptive switching control law as follows:

$$u_{\text{sw}_k} = -\mu_{N_k} \cdot \eta \cdot \frac{\text{sgn}(s_k)}{\Gamma_k}, \quad (22)$$

where μ_{N_k} is the adaptive adjusting factor depending on the output of RBF neural network. Thus, this parameter will be optimized using RBF neural network in the following section.

Therefore, we can get the adaptive neural network sliding mode control law as follows:

$$\begin{aligned} u_k &= u_{\text{eq}_k} + u_{\text{sw}_k} = \frac{1}{\Gamma_k} [-c_k \dot{e}_{(2k-1)} + \dot{x}_{d(2k)} - f_k(x, t) \\ &\quad - \mathfrak{R}_k] - \mu_{N_k} \cdot \eta \cdot \frac{\text{sgn}(s_k)}{\Gamma_k} = \frac{1}{\Gamma_k} [-c_k \dot{e}_{(2k-1)} \\ &\quad + \dot{x}_{d(2k)} - f_k(x, t) - \mathfrak{R}_k - \mu_{N_k} \cdot \eta \cdot \text{sgn}(s_k)]. \end{aligned} \quad (23)$$

Remark 2. The adaptive neural network sliding mode controller presented in (23) not only inherits the capacity of parameters optimization for M-CBDCS described in Section 3.2.1 but also can identify and estimate the modeling errors Δf , $\Delta \Gamma_k$ and composite disturbance for each channel \mathfrak{R}_k .

Proof. In order to demonstrate the stability of the designed controller, consider the following Lyapunov function candidate:

$$V = \frac{1}{2} s^2. \quad (24)$$

Based on (18) and (24), we can obtain the following differential equation:

$$\begin{aligned} \dot{V} &= s \dot{s} = s [c_k \dot{e}_{(2k-1)} - \dot{x}_{d(2k)} + f_k(x, t) + \Gamma_k u_k + \mathfrak{R}_k] \\ &= s \left[c_k \dot{e}_{(2k-1)} - \dot{x}_{d(2k)} + f_k(x, t) \right. \\ &\quad \left. + \Gamma_k \left(\frac{1}{\Gamma_k} [-c_k \dot{e}_{(2k-1)} + \dot{x}_{d(2k)} - f_k(x, t) - \mathfrak{R}_k \right. \right. \\ &\quad \left. \left. - \mu_{N_k} \cdot \eta \cdot \text{sgn}(s_k) \right) + \mathfrak{R}_k \right] = s (-\mu_{N_k} \cdot \eta \\ &\quad \cdot \text{sgn}(s_k)) = -\mu_{N_k} \cdot \eta \cdot |s|. \end{aligned} \quad (25)$$

If $\mu_{N_k} \geq 0$, we can get

$$\dot{V} = s \dot{s} \leq 0. \quad (26)$$

Therefore, the stability of adaptive neural network sliding mode controller designed in this paper can be guaranteed when μ_{N_k} , the output of RBF neural network, is used to adaptively adjust the switching control law of sliding mode. \square

3.2.1. Adaptive RBF Neural Network Based on M-CBDCS. For performance and applicability purposes, the M-CBDCS is chosen for optimizing the essential parameters of RBF network as described in Figure 3. There exist two kinds of communities according to its characteristics in M-CBDCS, where the community searching capacity is evaluated by the cooperation weights and response performance between different communities. Thus, considering the essential parameters optimization problem, the adaptive performance of RBF

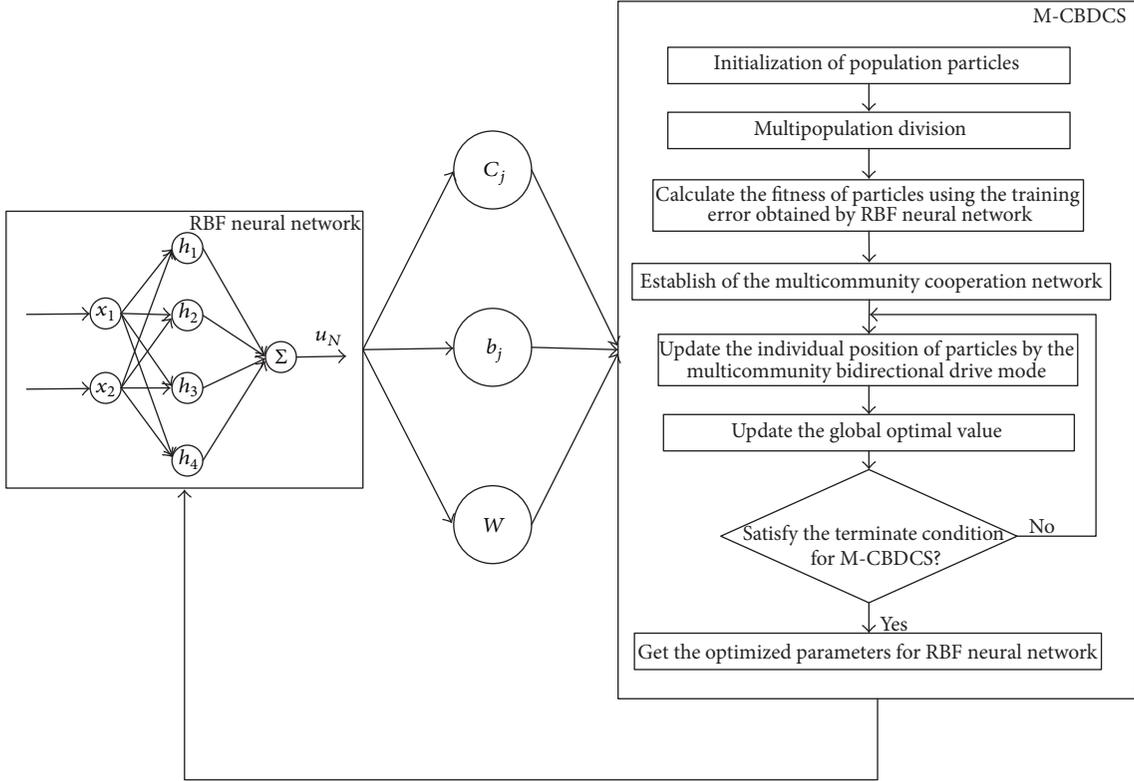


FIGURE 3: The flow diagram about the adaptive RBF neural network with M-CBDCS.

neural network is improved by guiding the evolution of the entire collaboration network from the community with powerful searching ability. Consequently, the center vector C_j , width vector b_j , and connection weights W are optimized online adaptively during the RBF neural network training procedure.

Note that RBF neural network has a great advantage of high speed learning and generalization [28, 29], which could realize the linearization of nonlinear problem through the transformation of hidden layer. Therefore, the RBF neural network is selected as an approach for switching law adaptive production expressed by (22).

The structure of RBF neural network is shown as follows.

The input layer $X = [x_1, x_2]^T$ is the vector of two inputs for this network. The input vector is designed to be two elements, which are the sliding mode s and its differential \dot{s} .

The hidden layer $H = [h_1, h_2, h_3, h_4]^T$ is the vector of activation functions in RBF neural network. The activation function chosen as the Gaussian function is as follows:

$$h_j = \exp\left(-\frac{\|X - C_j\|^2}{2b_j^2}\right) \quad j = 1, 2, \dots, 4, \quad (27)$$

where C_j is the center vector of this network and b_j is the base width vector of this network.

The output layer $\mu_N = \sum_{j=1}^4 \omega_j \cdot h_j = W^T \cdot H$ is used as the adaptive adjusting factor for sliding mode control.

$W = [\omega_1, \omega_2, \omega_3, \omega_4]$ is the vector of weighting factors which connect the hidden layer and the output layer of the network.

Remark 3. For the RBF neural network, the location of hidden unit is determined by the center vector and the size of control area is determined by the width vector. Moreover, the connection weights could directly influence the training time and the output of training as part of the network output. Therefore, the selection of essential parameters including the center vector, width vector, and connection weights is most important to improve the approximation performance of RBF neural network.

In this study a novel M-CBDCS algorithm is presented to uniformly update the essential parameters adaptively for RBF neural network with perfect convergence and high precision. The essential parameters of RBF neural network, including the center vector, width vector, and connection weights, are defined as the particle position vector in particle searching space. Noticeably, the communities are divided into model community (MC) and common community (CC) according to the advantages and disadvantages of community fitness, and the bidirectional interaction mode between communities is constructed as is shown in Figure 4. Under this mode, the member of common community CC_j will be selected into the model community and the worst community in MC will be abandoned according to the bidirectional evolutionary rules. Besides, the learning factor P_m is introduced

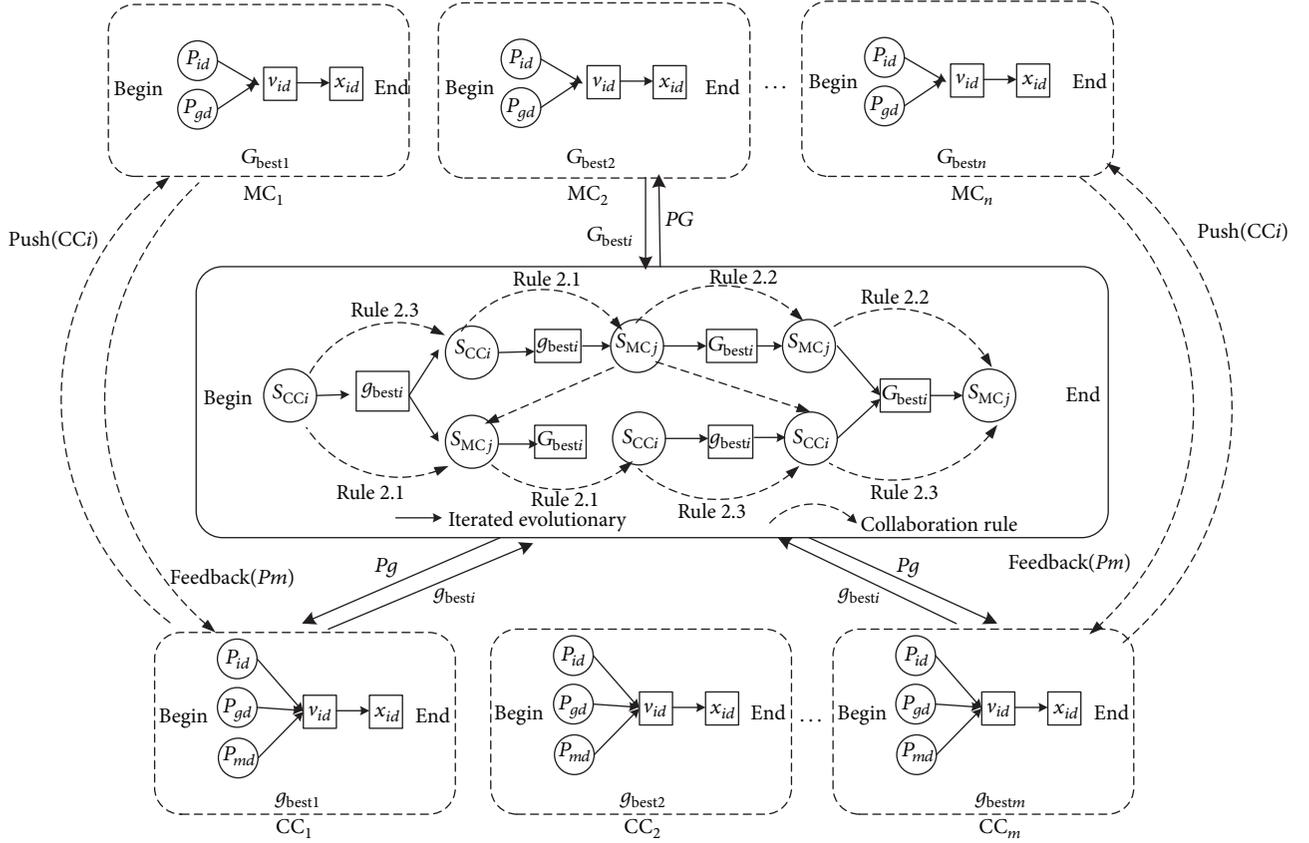


FIGURE 4: The multicommunity bidirectional drive mode.

into the evolution rules of the common communities to improve the searching ability. Thus, the asynchronous parallel implementation strategy for M-CBDCS is given, which could realize the parallel evolution within community and among communities. The designed searching strategy will benefit the overall optimization performance for RBF neural network.

Then, considering the influence of the network structure parameters on the validity of neural network approximating, the fitness function of parameters optimizing in M-CBDCS algorithm can be constructed as follows:

$$f = \frac{1}{N} \sum_{n=1}^N \sum_{m=1}^M (y_{mn} - \hat{y}_{mn})^2, \quad (28)$$

where y_{mn} is the practical output value of network after optimization; \hat{y}_{mn} is the desired output value of network; N is the number of training samples; M is the number of output neurons.

Particularly, the asynchronous parallel implementation strategy for RBF neural network is displayed in Figure 5. The M-CBDCS is thought to regulate the essential parameters adaptively, where the structure parameters of RBF neural network are used as initial space. The nonlinear approximation error is obtained and serves as a novel leakage term in the adaptations to guarantee the sliding surface convergence stably and estimate the chattering phenomenon. The detailed implementation can be seen in the following steps.

Step 1 (initialization). Determine the RBF neural network topology including the number of hidden layer and neurons for each layer.

An initial population is selected from the structure parameters of RBF neural network. Assume the total number of particles in population is n ; the number of communities is q ; the iteration times, acceleration coefficients, and the inertia coefficient are initial set.

Step 2. According to the number of communities, the initial particles are evenly distributed to the processes and the community size is $\text{int}(n/q)$ at present. The remaining particles are randomly assigned to processes after rounding.

Step 3. Calculate the fitness of each particle in communities by (28).

Step 4. Calculate the fitness value of populations and divide the populations into two groups according to the decision threshold: model community and common community. Then construct the multicommunity cooperation network.

Step 5. Update the position and velocity of particles according to the multicommunity bidirectional drive rules.

Step 6. Update the global optimal value from model communities.

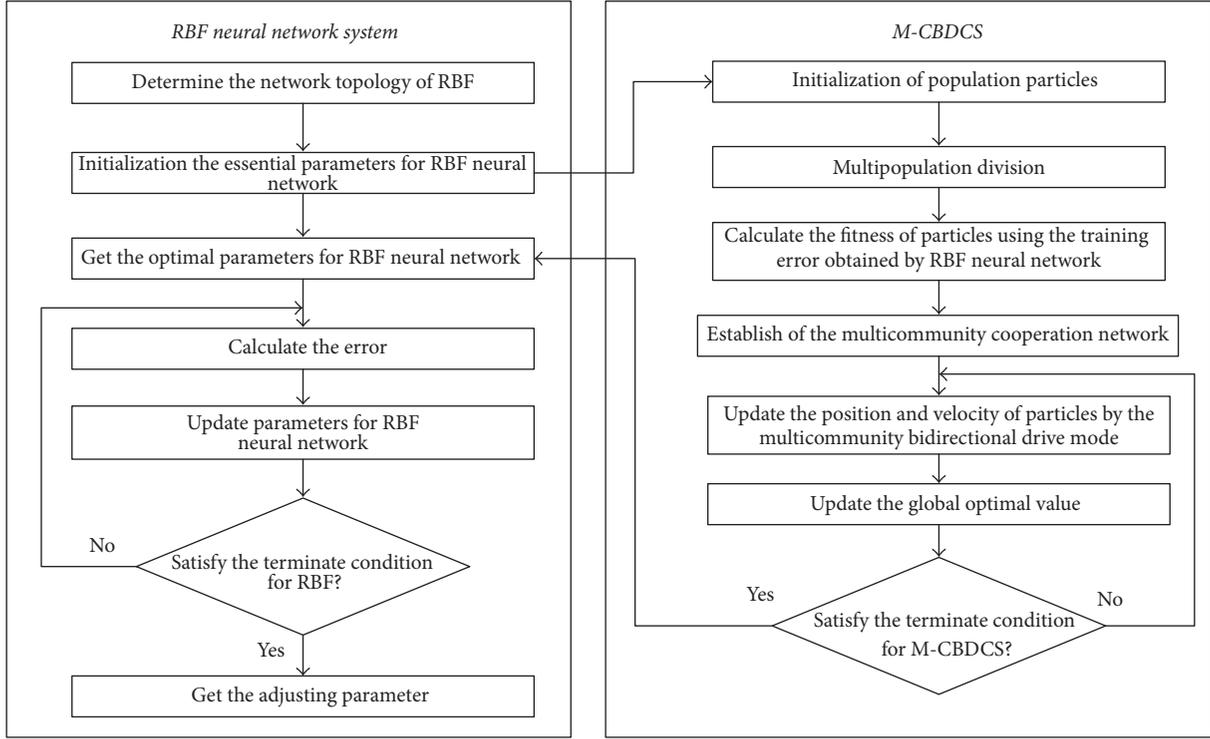


FIGURE 5: The flow diagram about the adaptive RBF neural network with the optimization of M-CBDCS.

Step 7. Loop to the Step 5 until the iterative termination condition of M-CBDCS is met.

Step 8. Get the optimal parameters for RBF neural network from M-CBDCS.

Step 9. Calculate the error between the actual output value and the desired output value of network.

Step 10. Update the parameters for RBF neural network according to the error above.

Step 11. Loop to Step 9 until the iterative termination condition of RBF neural network is met. And we can get the adaptive adjusting parameter for sliding mode control.

Figure 5 displays the flow chart of the adaptive RBF neural network under the optimization of M-CBDCS.

3.2.2. Adaptive Neural Network Sliding Mode Control for QTRA Attitude. The block diagram of the designed adaptive neural network sliding mode control structure is shown in Figure 6. In this diagram, the advantage of M-CBDCS is to adjust the value of the three key parameters adaptively, which can enhance the ability of nonlinear approximation and accelerate the convergence speed for RBF neural network. Similarly, the adaptive RBF neural network is used to generate the adaptive adjustment factor in sliding mode control to reduce the chattering phenomenon.

TABLE 1: Parameters for experiment.

Parameter	Value
Mass for QTRA (kg)	6.25
I_{xx} (kg·m ²)	4.568
I_{yy} (kg·m ²)	3.109
I_{zz} (kg·m ²)	2.796
I_{xz} (kg·m ²)	0.028

4. Simulations

In this section, in order to verify the effectiveness of the controller design algorithm proposed in this paper, simulations by using Matlab/Simulink for three control schemes are performed: (1) the adaptive neural network sliding mode controller (ANNSMC) proposed in this paper; (2) the classical neural network sliding mode controller (CNNSMC) proposed in [30]; (3) the dynamic inverse PID controller (DIPID) proposed in [31].

The designed controller is applied to the attitude control of the QTRA with respect to different reference signals: step signal, random noise, external disturbances, and hybrid superposition signal. And the parameters for the experiment are shown in Table 1.

Case 1 (step response). Considering that the landing process of QTRA usually adopts the method of equal angle descent,

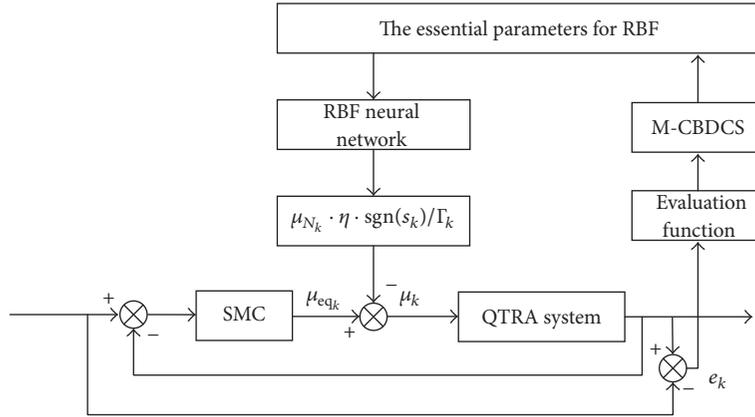


FIGURE 6: The control structure of the adaptive neural network sliding mode controller for QTRA.

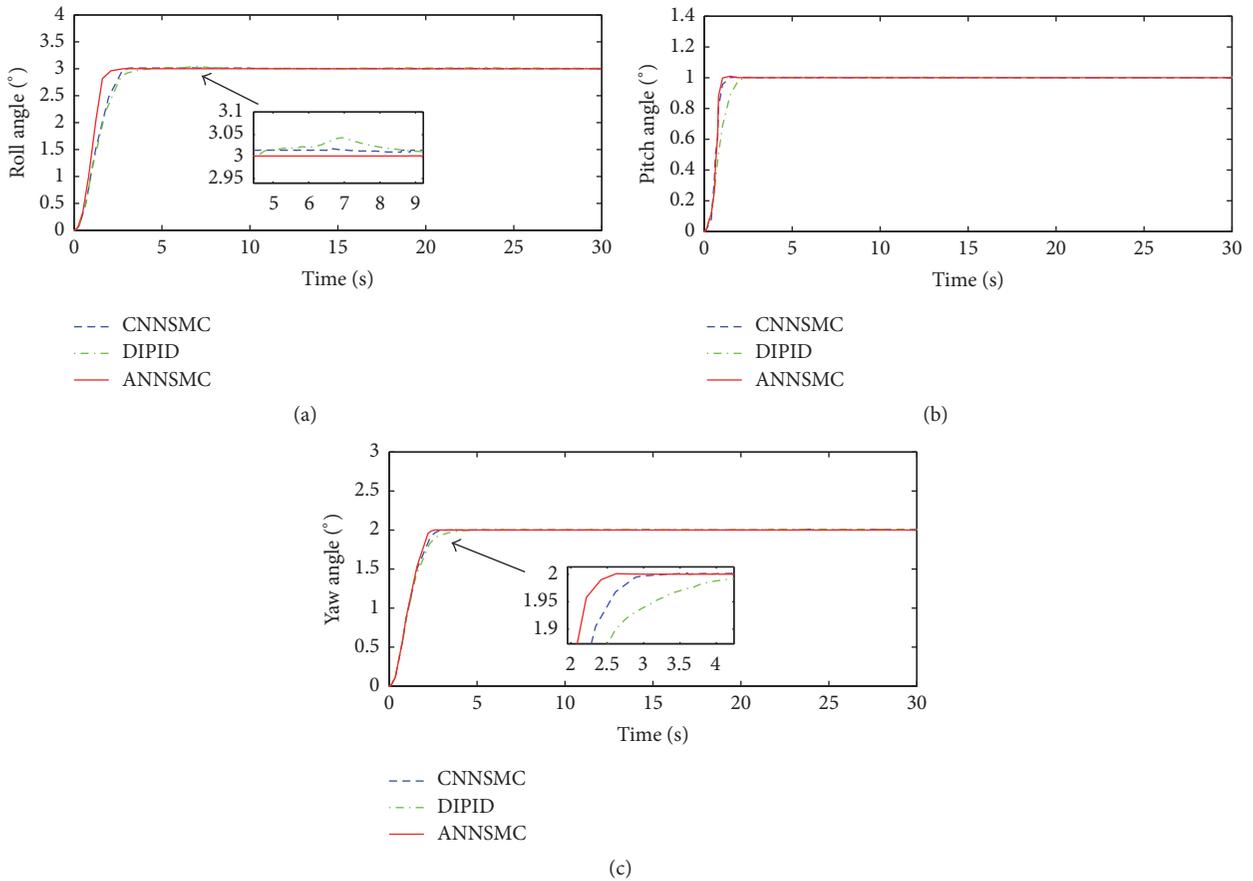


FIGURE 7: Helicopter attitude behavior response to the step signal.

the step signal is used to verify the control performance of these controllers for steady tracking in the simulation. We start the aircraft at the attitude $(\phi, \theta, \varphi) = (0, 0, 0)$ and the aircraft has the task of performing hover flight at $(\phi, \theta, \varphi) = (3^\circ, 1^\circ, 2^\circ)$. The evolution and convergence of states (ϕ, θ, φ) to the desired reference with the initial conditions are shown in Figure 7.

The results show that the method presented in this paper can eliminate steady state errors effectively with the fast

convergence speed. But the other methods about CNNSMC and DIPID have a certain fluctuation in the process of tracking instruction with poor steady state performance.

Case 2 (random noise). Normally this simulation is designed to observe the effects of a white Gaussian random noise, which is generated with a maximum amplitude of 0.01. In this case, the disturbance in the feedback loops is to simulate the sensor noise. And the simulation results show the comparison of adaptive controller versus other controllers

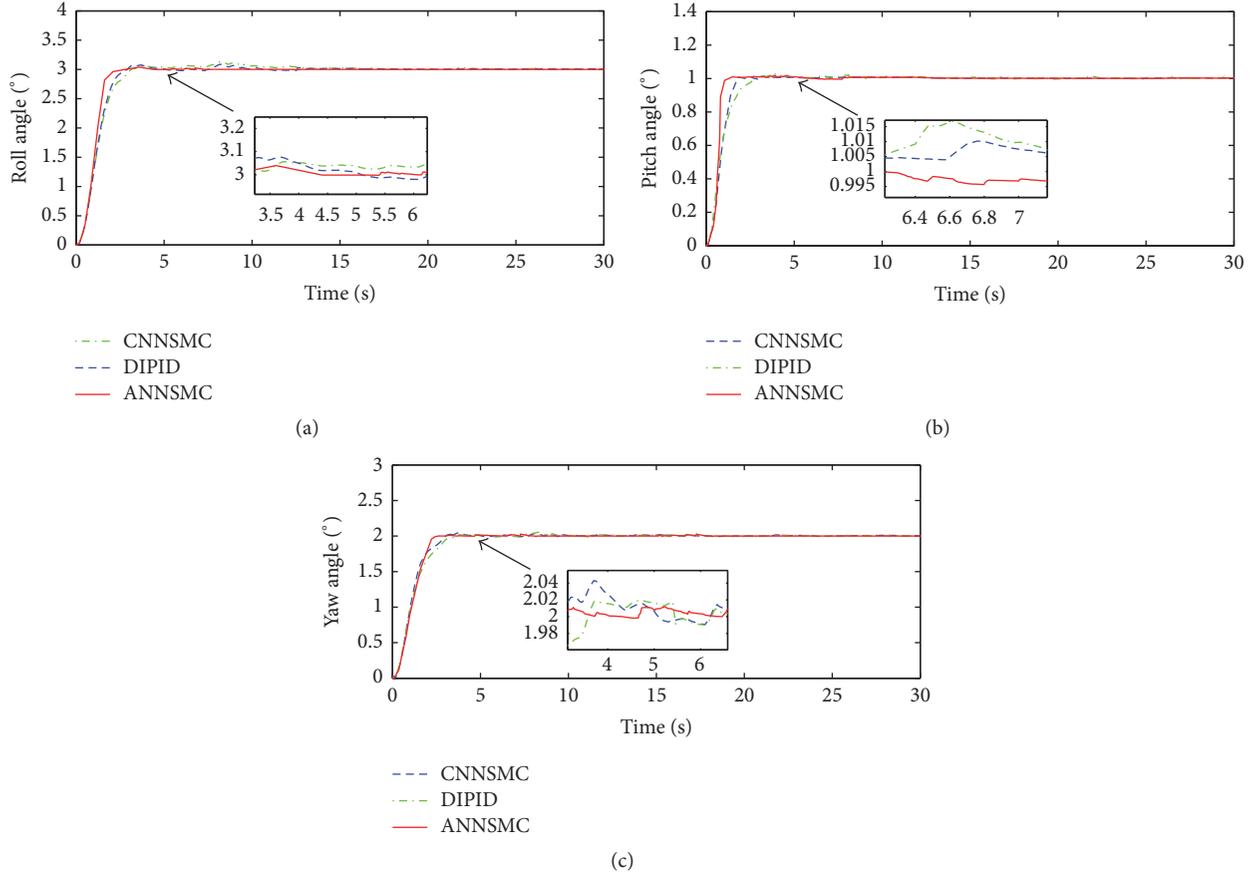


FIGURE 8: Helicopter attitude behavior response to the step signal with random noise.

in Figure 8, which implied that the ANNSMC controller has a better capability of suppressing noise than the other two methods. It can be concluded that the adaptive learned upper bound of noise can reduce chattering significantly with the introduction of M-CBDCS. And then a small boundary layer about switching surface can be generated by using the adaptive sliding mode control. Consequently, the ANNSMC controller gives a smooth control and proves to be effective for attitude control of QTRA.

Case 3 (external disturbances). In the actual system, the parameter perturbation and external disturbance are inevitable. In order to further verify the anti-interference performance of the ANNSMC controller, the external disturbances test is carried in this section. As we can see from Figure 9, the capacity of the presented controller guarantees a more steady flight of the airframe.

Case 4 (dynamic tracking). To verify the tracking capacity for the proposed method, one more simulation is run such that the desired reference is given using a hybrid superposition signal. The simulation results indicate that the maximum tracking error percentage in the attitude of QTRA is approximately 1.89% as shown in Figure 10. It is important to note that angle references are tracked with negligible steady state errors.

In summary, the control strategy proposed in this paper has good performance for QTRA attitude control. Results in Case 1 show that the presented control strategy has relatively lesser settling time in the attitude control performance compared to other controllers. And Cases 2, 3, and 4 further verify the robustness and stability of the designed controller.

5. Conclusion

This paper describes an adaptive online control of attitude angles for nonlinear QTRA systems. The attitude dynamics of this aircraft including uncertain external disturbance are derived at VTOL mode. The proposed control strategies were evaluated by appropriate comparisons showing satisfactory results. A novel RBF neural network sliding mode control system was designed and built to implement an attitude control law. The presented adaptive RBF neural network is based on a multicommunity bidirectional drive collaborative search algorithm since it is suitable for implementation purposes with powerful parallel computing capacity. The comparative simulations between the proposed control method and classical neural network sliding mode control and dynamic inverse PID control were designed to simulate an attitude stabilized flight in VTOL stage with external disturbances.

Future work includes the transition flight between cruise and hover flight modes to evaluate the accuracy of the control

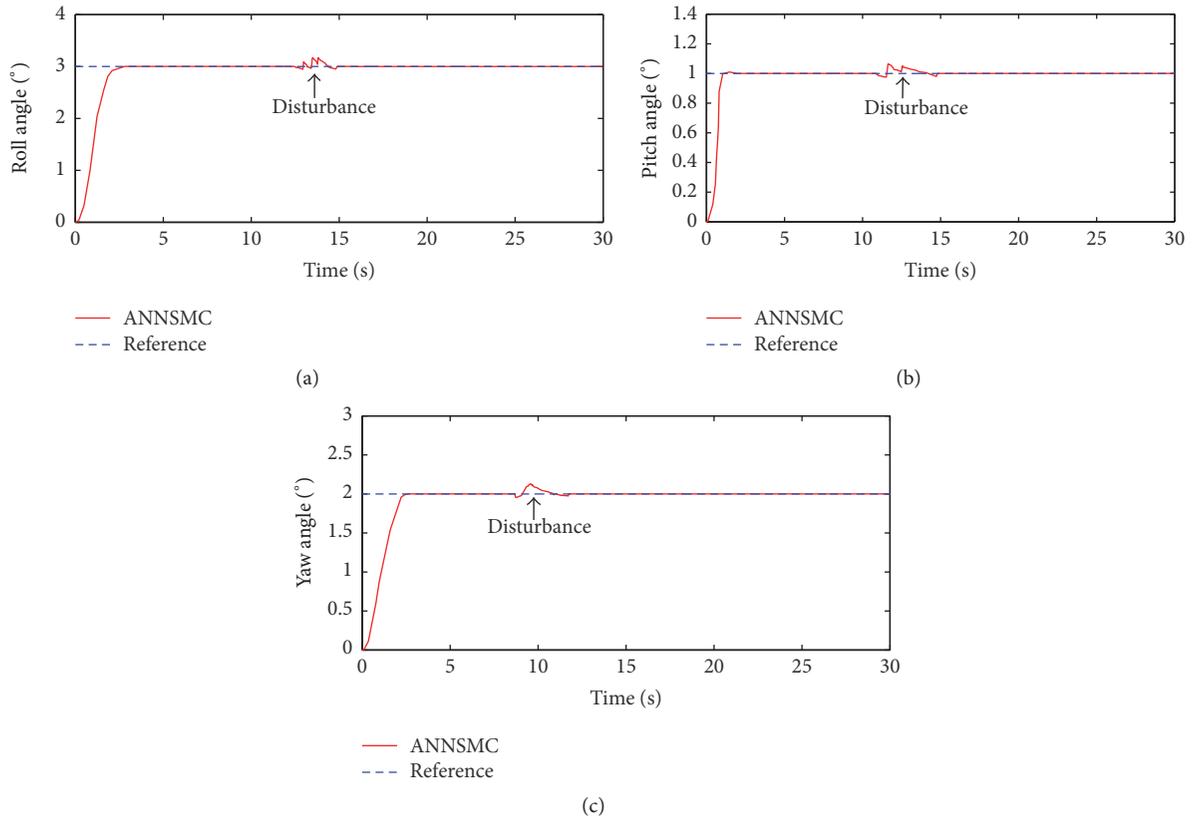


FIGURE 9: Helicopter attitude behavior response to the step signal with external disturbances.

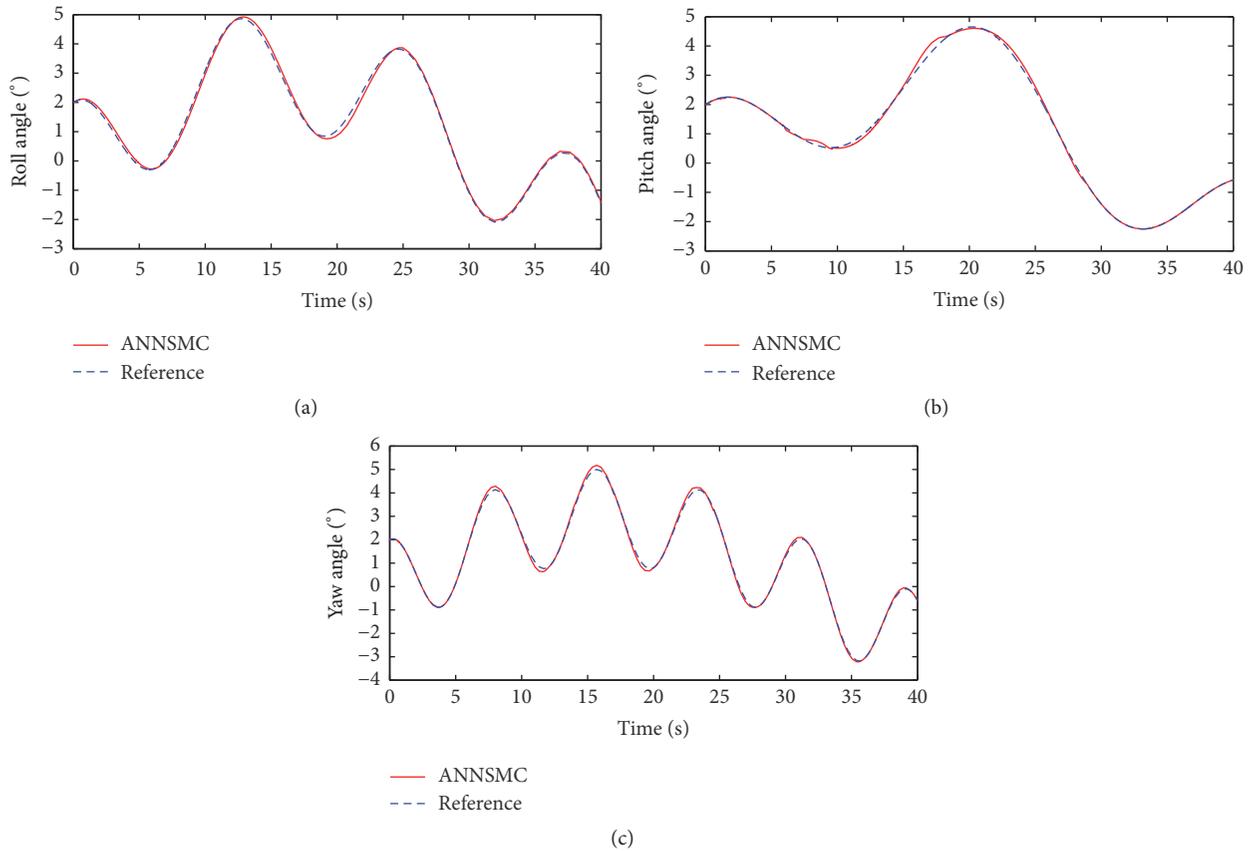


FIGURE 10: Helicopter attitude behavior response to the hybrid superposition signal.

model to validate the control algorithm and the development of experimental convertible aircraft to implement the proposed design and configuration.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This paper was supported by the National Natural Science Foundation of China (51365022) and the Science Research Foundation Project of Education Department of Yunnan Province (2016YJS022).

References

- [1] S. Abdulkerim Fatih and e. Erdinc, "Erdinc, altugsystem, design of a novel tilt-roll rotor quadrotor UAV," *Journal of Intelligent Robotic Systems*, vol. 84, Article ID 575599, pp. 575–599, 2016.
- [2] X. Wang, "Takeoff/landing control based on acceleration measurements for VTOL aircraft," *Journal of the Franklin Institute*, vol. 350, no. 10, pp. 3045–3063, 2013.
- [3] T. Oktay and C. Sultan, "Modeling and control of a helicopter slung-load system," *Aerospace Science and Technology*, vol. 29, no. 1, pp. 206–222, 2013.
- [4] C. S. Lee and R. V. Gonzalez, "Fuzzy logic versus a PID controller for position control of a muscle-like actuated arm," *Journal of Mechanical Science and Technology*, vol. 22, no. 8, pp. 1475–1482, 2008.
- [5] J. Na, G. Herrmann, and K. Zhang, "Improving transient performance of adaptive control via a modified reference model and novel adaption," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 8, pp. 1351–1372, 2017.
- [6] J. Li and Y. Li, "Dynamic analysis and PID control for a quadrotor," in *Proceedings of International Conference on Mechatronics and Automation, (ICMA '11)*, pp. 573–578, August 2011.
- [7] A. Das, F. Lewis, and K. Subbarao, "Backstepping approach for controlling a quadrotor using lagrange form dynamics," *Journal of Intelligent and Robotic Systems*, vol. 56, no. 1, pp. 127–151, 2009.
- [8] A. Scottedward Hodel and C. E. Hall, "Variable-structure PID control to prevent integrator windup," *IEEE Transactions on Industrial Electronics*, vol. 48, no. 2, pp. 442–451, 2001.
- [9] H.-B. Shin and J.-G. Park, "Anti-windup PID controller with integral state predictor for variable-speed motor drives," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 3, pp. 1509–1516, 2012.
- [10] C. Peng, Y. Bai, G. Qiao, X. Gong, and Y. Tian, "Anti-windup and multi-mode PID control of yaw movement for a quad-rotor UAV," *Robot*, vol. 37, no. 4, pp. 415–423, 2015.
- [11] J.-B. Song, Y.-S. Byun, J. Kim, and B.-S. Kang, "Guidance and control of a scaled-down quad tilt prop PAV," *Journal of Mechanical Science and Technology*, vol. 29, no. 2, pp. 807–825, 2015.
- [12] J. Na, M. N. Mahyuddin, G. Herrmann, X. Ren, and P. Barber, "Robust adaptive finite-time parameter estimation and control for robotic systems," *International Journal of Robust and Nonlinear Control*, vol. 25, no. 16, pp. 3045–3071, 2015.
- [13] J.-J. Xiong and E.-H. Zheng, "Position and attitude tracking control for a quadrotor UAV," *ISA Transactions*, vol. 53, no. 3, pp. 725–731, 2014.
- [14] B.-T. Zhang, F.-R. Gao, and K. Yao, "Neural network and adaptive algorithm-based fractional order sliding mode controller," *Control Theory and Applications*, vol. 33, no. 10, pp. 1373–1377, 2016.
- [15] R.-J. Wai, "Fuzzy sliding-mode control using adaptive tuning technique," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 1, pp. 78–87, 2007.
- [16] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, pp. 2004–2016, 2014.
- [17] J. Na, Q. Chen, X. Ren, and Y. Guo, "Adaptive prescribed performance motion control of servo mechanisms with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 486–494, 2014.
- [18] Y.-A. Zhang, M.-Z. Yu, and H.-L. Wu, "Sliding mode synchronization of fractional-order chaotic systems based on adaptive neural network," *Control and Decision*, vol. 30, no. 5, pp. 882–886, 2015.
- [19] P. Li, J. J. Ma, W. Q. Li, and Z. Q. Zheng, "Improved integral sliding mode control for a class of nonlinear uncertain systems," *Control and Decision*, vol. 24, no. 10, pp. 1463–1472, 2009.
- [20] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [21] S. Seshagiri and H. K. Khalil, "Robust output feedback regulation of minimum-phase nonlinear systems using conditional integrators," *Automatica*, vol. 41, no. 1, pp. 43–54, 2005.
- [22] J. H. Wang and J. B. Hu, "Adaptive backstepping higher-order terminal sliding mode control for uncertain nonlinear systems," *Control and Decision*, vol. 27, no. 3, pp. 413–418, 2012.
- [23] K. Mathiyalagan and K. Balachandran, "Finite-time stability of fractional-order stochastic singular systems with time delay and white noise," *Complexity*, vol. 21, no. S2, pp. 370–379, 2016.
- [24] S. Li, H. Du, and X. Yu, "Discrete-time terminal sliding mode control systems based on Euler's discretization," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 5, no. 2, pp. 546–552, 2014.
- [25] J. Zhu, S. Yakun, L. Hongyi, and L. Li, "Tracking control of uncertain switched nonlinear cascade systems: a nonlinear H_∞ sliding mode control method," *Nonlinear Dynamics*, vol. 73, no. 8, pp. 1803–1812, 2014.
- [26] Y. Yin, H. Niu, and B. Chang, "Hongwei Niu, BinLei Chang, Multi-community bidirectional drive collaborative search algorithm," *Computer Integrated Manufacturing Systems*, vol. 23, no. 7, pp. 1581–1592, 2017.
- [27] Y. Yin and Y. Huang, "Method of product innovation acquisition based on dynamic response particle swarm optimization," *Advanced Materials Research*, vol. 542-543, pp. 3–7, 2012.
- [28] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [29] A. F. Şenkul and E. Altuğ, "System design of a novel Tilt-roll rotor quadrotor UAV," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 84, pp. 1–25, 2016.

- [30] H. J. Jayakrishnan, "Position and attitude control of a quadrotor UAV using super twisting sliding mode," *IFAC-PapersOnLine*, vol. 49, no. 1, pp. 284–289, 2016.
- [31] A. Alkamachi and E. Erçelebi, "Modelling and genetic algorithm based-PID control of H-shaped racing quadcopter," *Arabian Journal for Science & Engineering*, pp. 1–10, 2017.

Research Article

The Dissolved Oxygen Prediction Method Based on Neural Network

Zhong Xiao,¹ Lingxi Peng,¹ Yi Chen,² Haohuai Liu,³ Jiaqing Wang,¹ and Yangang Nie⁴

¹School of Mechanical and Electrical Engineering, Guangzhou University, Guangzhou 510006, China

²School of Business Administration, Guangzhou University, Guangzhou 510006, China

³School of Chemistry, Guangzhou University, Guangzhou 510006, China

⁴School of Education, Guangzhou University, Guangzhou 510006, China

Correspondence should be addressed to Lingxi Peng; scu.peng@gmail.com and Yangang Nie; niezi66@21cn.com

Received 3 July 2017; Accepted 29 August 2017; Published 9 October 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Zhong Xiao et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The dissolved oxygen (DO) is oxygen dissolved in water, which is an important factor for the aquaculture. Using BP neural network method with the combination of purelin, logsig, and tansig activation functions is proposed for the prediction of aquaculture's dissolved oxygen. The input layer, hidden layer, and output layer are introduced in detail including the weight adjustment process. The breeding data of three ponds in actual 10 consecutive days were used for experiments; these ponds were located in Beihai, Guangxi, a traditional aquaculture base in southern China. The data of the first 7 days are used for training, and the data of the latter 3 days are used for the test. Compared with the common prediction models, curve fitting (CF), autoregression (AR), grey model (GM), and support vector machines (SVM), the experimental results show that the prediction accuracy of the neural network is the highest, and all the predicted values are less than 5% of the error limit, which can meet the needs of practical applications, followed by AR, GM, SVM, and CF. The prediction model can help to improve the water quality monitoring level of aquaculture which will prevent the deterioration of water quality and the outbreak of disease.

1. Preface

Dissolved oxygen [1] refers to the amount of oxygen dissolved in water, usually expressed in DO. It is an important indicator to study the water self-purification ability. The level of DO directly affects the aquaculture's food intake, feed conversion rate, and disease resistance. Low DO or hypoxic water environment has a great impact on aquaculture organisms. As an example, if the aquaculture shrimp is often raised in the low DO water environment, it will intake little food, which will lead to low conversion rate of food, slow growth, and low disease resistance. The hypoxia can directly or indirectly lead to a large number of live shrimp.

Based on the study of the lowest time of DO in aquaculture water and the analysis of the daily variation of low DO values, we can estimate and forecast its development trend. The basis for decision-making is provided to improve water quality for preventing water quality deterioration with the low DO. It helps to control and reduce aquaculture risk. At

present, the common prediction models include curve fitting (CF) [2], autoregression (AR) [3], neural network (NN) [4–8], grey models (GM) [9, 10], support vector machine (SVM) [11, 12], and other models [13]. But there is no literature on the comparison and experimentation of these methods. Therefore, in order to study an accurate and practical method of DO prediction, in this paper, NN method, CF method, AR method, GM method, and SVM method are compared using the 10 consecutive days of aquaculture data of southern China traditional aquaculture base in three shrimp culture ponds in Beihai, Guangxi Province. The experimental results show that the accuracy of NN prediction is high, and all the predicted values are less than 5%, which is the maximum acceptable predicted error rate. So the method can meet the needs of practical applications.

2. Artificial Neural Network

Artificial neural network [14, 15] is a biomimetic intelligent information processing system that simulates the structure

and function of human brain's neural network. It is suitable for describing complex nonlinear mapping relationships and has a strong ability to integrate information and fault tolerance. It has very good self-adaptability for nonregular, nonlinear structures. It is an important research method for computational intelligence.

Backpropagation neural network is currently the most widely used neural network. In 1986, it was proposed by Rumelhart and McClelland and others. It is a multilayer feed-forward neural network trained by error backpropagation algorithm. It solves the problem of the hidden layer connection of multilayer neural networks systematically and gives the complete derivation in mathematics. A multilayer feed-forward network using this algorithm for error correction is called a BP network. BP neural network has complex pattern classification ability and excellent multidimensional function mapping ability, which solves the Exclusive OR (XOR) and some other problems that simple sensor cannot. Structurally, the BP network has an input layer, a hidden layer, and an output layer. In essence, the BP algorithm is based on the square of the network error as the objective function, using the gradient descent method to calculate the minimum value of the objective function.

The BP neural network used in this paper has three layers, namely, L_A layer, input layer, L_B layer, hidden layer, and L_C layer, output layer. These layers are fully interconnected, and there is no connection between neurons within each hierarchy. L_A layer has n neurons, L_B layer has p neurons, L_C layer has m neurons, v_{hi} is the weight between the neuron h on the L_A layer and the neuron i on the L_B layer, and w_{ij} is the weight between the neurons i on the L_B layer and the neurons j on the L_C layer. The following is the BP network specific learning algorithm:

(1) Randomly initialize the weight of the BP network, and then a smaller random number is generally taken.

(2) Obtain the input vector $X = (x_1, x_2, \dots, x_n)$ and its corresponding expected output vector $T = (t_1, t_2, \dots, t_m)$ from the learning sample.

(3) Calculate the output values of neurons on the L_B layer according to (1), where $i = 1, 2, \dots, p$. The function f is an S-shaped function.

$$b_i = f\left(\sum_{h=1}^n x_h v_{hi}\right). \quad (1)$$

(4) Calculate the output values of neurons on the L_C layer according to (2), where $j = 1, 2, \dots, m$.

$$c_j = f\left(\sum_{i=1}^p b_i w_{ij}\right). \quad (2)$$

(5) Calculate the generalized error of the neurons in the L_C layer according to (3), where $j = 1, 2, \dots, m$.

$$d_j = (t_j - c_j) f'\left(\sum_{i=1}^p b_i w_{ij}\right) = (t_j - c_j) c_j (1 - c_j). \quad (3)$$

(6) Calculate the error of L_B layer neurons relative to each d_j according to (4), where $j = 1, 2, \dots, m$. This step

propagates the error of the L_C layer's neurons back to the L_B layer.

$$e_i = f'\left(\sum_{h=1}^n x_h v_{hi}\right) \sum_{j=1}^m w_{ij} d_j = b_i (1 - b_i) \sum_{j=1}^m w_{ij} d_j. \quad (4)$$

The activation function of neurons must be derivable everywhere in the BP algorithm, so the common applications in the BP network are purelin, logsig, and tansig. It has the following properties:

$$\text{purelin}(x) = x,$$

$$\text{purelin}'(x) = 1,$$

$$\log \text{sig}(x) = \frac{1}{1 + e^{-x}},$$

$$\log \text{sig}'(x) = \frac{1}{1 + e^{-x}} - \frac{1}{(1 + e^{-x})^2} = f(x) - f^2(x), \quad (5)$$

$$\tan \text{sig}(x) = \frac{1}{1 + \arctan(x)},$$

$$\tan \text{sig}'(x) = 1 - \tan \text{sig}(x)^2.$$

In our experiments, we use the combination of these three activation functions, which greatly improve the prediction accuracy, and a large amount of calculation has been reduced.

(7) Adjust the weights between the L_B and L_C layers' neurons according to (6), where $i = 1, 2, \dots, p$, $j = 1, 2, \dots, m$, and η is the learning rate.

$$\Delta w_{ij} = \eta b_i d_j. \quad (6)$$

(8) Adjust the weights between the L_A and L_B layers' neurons according to (7), where $h = 1, 2, \dots, n$, $i = 1, 2, \dots, p$.

$$\Delta v_{hi} = \eta x_h e_i. \quad (7)$$

(9) Go to step (2) until the error d_j becomes small enough or becomes zero for $j = 1, 2, \dots, m$.

Artificial neural network model has the advantages of autonomous learning, information memory, knowledge reasoning, optimization calculation, and nonlinear fitting ability. It is able to predict data after learning and training. But it is not suitable for small samples and high-dimensional data. Its shortcomings are mainly the large amount of computing, slow convergence rate, being easy to fall into the local minimum, difficulty in determining the number of hidden layer's neurons, and so on.

3. Experiments

Beibu Gulf is located in the southern part of Beihai, Guangxi Province, which is rich in marine resources. It is a traditional aquaculture base in southern China. It is one of the four major fishing grounds in China. In 2016, its aquaculture area has been more than 50 million acres. The DO measurements were taken from the representative three shrimp farming ponds in

Beihai, Guangxi province. As a result, the DO value of the first pond was deteriorating at 4:00 am, but the DO values of the second and the third ponds were normal.

As far as the parameter is concerned, the order of curve fitting is 2. The autoregression (AR) and grey models (GM) need no parameters. The parameters used in the neural network (NN) were set to the number of iterations of 5000, the learning rate of 0.01, and the target value of 0.000000001. The support vector machine (SVM) uses different penalty coefficient and nuclear function according to the training data to minimize the training error.

The DO value in aquaculture water is susceptible to solar radiation, temperature, pressure, wind speed, and other environmental factors. It is cyclical over time. Generally it reaches the highest value between 15:00 and 17:00 and reaches the lowest value between 3:00 and 5:00. The lowest value of DO is an important indicator of water pollution. Therefore, the test was conducted for study based on the DO values at 4:00 am for seven consecutive days. By using five models, CF, AR, NN, SVM, and GM, the DO values at 4:00 am for the eighth day, the ninth day, and the tenth day were analyzed; the values of the low dissolved oxygen period were predicted. This will be able to predict the development trend of DO in water. Based on this, it is possible to prevent deterioration of water quality and prevent disease outbreaks in advance.

The total DO values were selected at 4:00 am in mid-June for seven consecutive days. The sunshine duration of the seven days is about 16 hours. The sun is plenty. The average temperature is about 31 degrees. Day water temperature difference is ± 3 degrees. The average humidity is about 90%. The average pressure is 1 atmosphere. The wind level is 2 to 3. There is no typhoon and rainstorm. The baits are delivered at three times a day at 8:00 am, 12:00 noon, and 18:00 pm. The aerator runs twice a day, once in the afternoon, 14:00 to 17:00, and once between 3:00 and 5:00. The test assumes that other conditions have no effect on DO.

3.1. The First Experiment. The shrimp culture pond is located in Hepu County Party Town, Beihai, Guangxi Province, China. It covers an area of 7.5 acres. The average water depth is 1.4 meters. It breeds South American white shrimp with 60000 shrimps per acre. It planted the *Hydrilla verticillata*, tape grass, and so forth.

The actual value of the output and DO of each model is shown in Figure 1. The absolute error between the predicted value and the actual value of the DO for the next 3 days is shown in Figure 2. As can be seen from Figure 2, the actual value of DO on day 8 is 3.8. The prediction data of each prediction model are accurate. The prediction accuracy from high to low is as follows: NN, CF, SVM, GM, and AR. The error of NN model is the smallest: it is only 0.005%. The error of GM model is the largest: it is 1.34%. They are all less than 5%. Therefore, this is fully in line with forecast requirements. The actual value of DO on day 9 is 3.7. The prediction accuracy from high to low is as follows: NN, CF, SVM, GM, and AR. The error of AR model is 5.25 and the rest are all less than 5%. The actual value of DO on day 10 is 3.5. The prediction accuracy from high to low is as follows: NN, AR, CF, SVM,

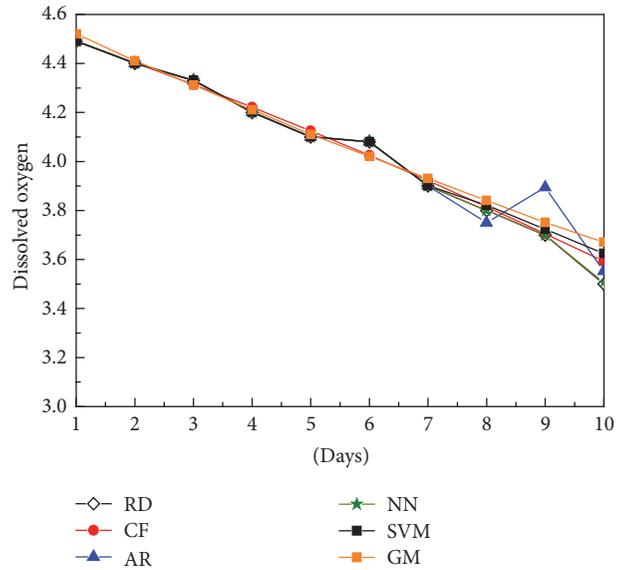


FIGURE 1: The actual value and predicted value for the first pond.

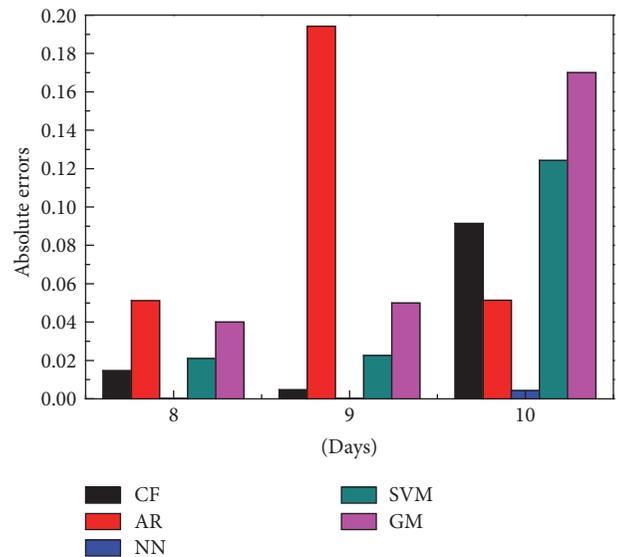


FIGURE 2: The absolute error between the actual and predicted values for the first pond.

and GM. The error of all models is less than 5%. This meets the forecast requirements.

Thus, the above five predictive models (CF, AR, NN, SVM, and GM) can meet the design requirements based on ignoring the fact that the error of AR model on day 9 is slightly larger. They can make accurate prediction of the next 3 days' DO at 4:00 am, which is the lowest value of DO in each day. Through the early warning of the deterioration of water quality, the amount of water in the pond is regulated. Thereby the water quality of the pond is improved and the aquaculture risk is reduced. In fact, the water quality of this shrimp pond is deteriorating because the density of the pond is too large and the regulation of water is not timely. Eventually, the serious

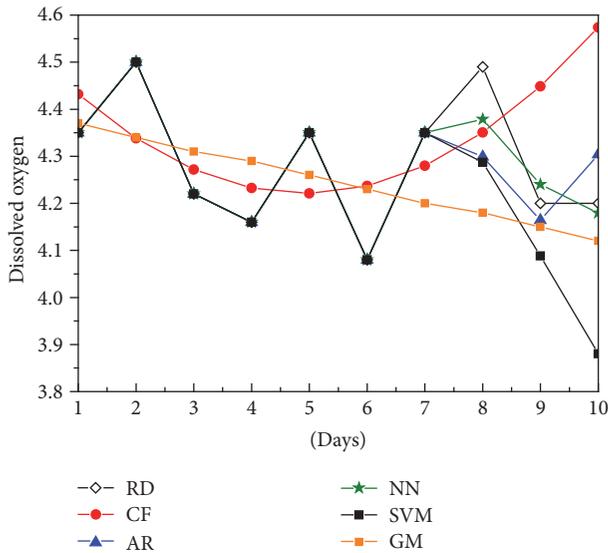


FIGURE 3: The actual value and predicted value for the second pond.

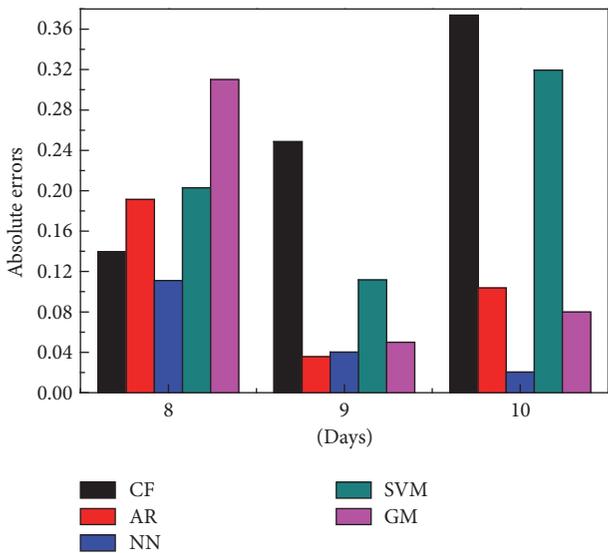


FIGURE 4: The absolute error between the actual and predicted values for the second pond.

accident broke out and nearly half of the shrimps died. It caused serious economic losses to aquaculture households.

3.2. The Second Experiment. The shrimp culture pond is located in Hepu County West Town, Beihai, Guangxi Province, China. It covers an area of 4.5 acres. The average water depth is 1.2 meters. It breeds South American white shrimp with 40000 shrimps per acre. It planted the waterweed, tape grass, and so forth.

The actual value of the output and DO of each model is shown in Figure 3. The absolute error between the predicted value and the actual value of the DO for the next 3 days is shown in Figure 4. As can be seen from Figure 4, the actual value of DO on day 8 is 4.49. The prediction accuracy of each

model from high to low is as follows: NN, CF, AR, SVM, and GM. The error of GM model is the largest: it is 6.904%. It is slightly higher than the 5%. The actual value of DO on day 9 is 4.2. The prediction accuracy of each model from high to low is as follows: AR, NN, GM, SVM, and CF. The error of CF model is the largest: it is 5.391%. The error of other models is less than 5%. The actual value of DO on day 10 is 4.2. The prediction accuracy of each model from high to low is as follows: NN, GM, AR, SVM, and CF. The error of SVM model is 7.607% and the error of CF model is 8.902%. They are higher than 5%. However, the others meet the forecast requirements.

The shrimp delivery of the pond did not exceed the standard. The lowest value of DO is only fluctuating up and down within a certain range. It can be seen that the current minimum DO fluctuation is not very large. The error between the predicted value and the actual value of DO is not large. Although the prediction error of SVM model and CF model in the third day is slightly larger, the prediction accuracy of NN model and AR model is still maintained at a high level.

3.3. The Third Experiment. The shrimp culture pond is located in Hepu County Party Town, Beihai, Guangxi Province, China. It covers an area of 8.5 acres. The average water depth is 1.1 meters. It breeds South American white shrimp with 42000 shrimps per acre. It planted the waterweed, *Hydrilla verticillata*, tape grass, and so forth. The actual value of the output and DO of each model is shown in Figure 5. The absolute error between the predicted value and the actual value of the DO for the next 3 days is shown in Figure 6. As can be seen from Figure 6, the actual value of DO on day 8 is 4.08. The prediction accuracy from high to low is as follows: NN, GM, SVM, AR, and CF. The error of CF model is the largest: it is only 7.458%. It is higher than 5%. The actual value of DO on day 9 is 4.2. The prediction accuracy from high to low is as follows: AR, NN, GM, SVM, and CF. Except that the error between the predicted value and the actual value of DO of CF model is 17.46%, the prediction error of other models is less than 5%. The actual value of DO on day 10 is 4.3. The prediction accuracy from high to low is as follows: NN, GM, AR, SVM, and CF. The prediction error of the models is less than 5% except the prediction error of NN model.

The shrimp delivery of the pond did not exceed the standard. The lowest value of DO is only fluctuating up and down within a certain range. The error between the predicted value and the actual value of DO is less than 5%. Among the models, the prediction accuracy of NN model is the highest.

In summary, three ponds were raised with the white leg shrimps. There are almost the same test conditions. From the comparison between the predicted values and the actual values of the three ponds, we can see that NN model has the highest accuracy of prediction. The prediction error rate of NN model is less than 5%. Among all the prediction values of OD, the maximum prediction error only appears on the eighth day of the second experiment. It is only 2.474%. The AR model comes next. There are two prediction error rates of test values exceeding 5%. But they are 5.25% and 6.58%. The prediction error rate is not much higher. GM model is ranked third, and SVM model is ranked fourth. Although

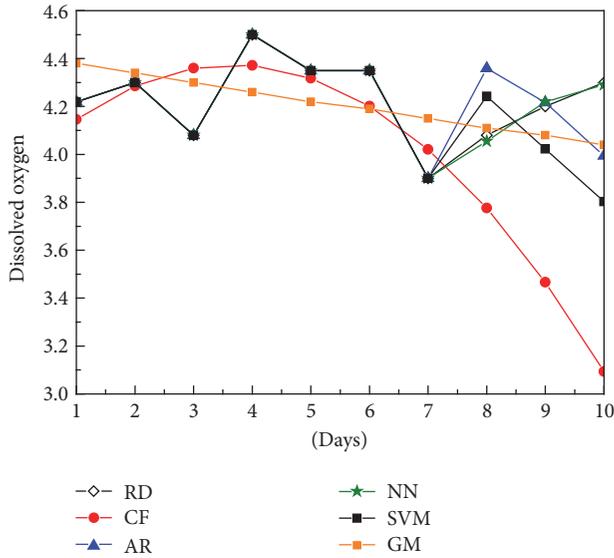


FIGURE 5: The actual value and predicted value for the third pond.

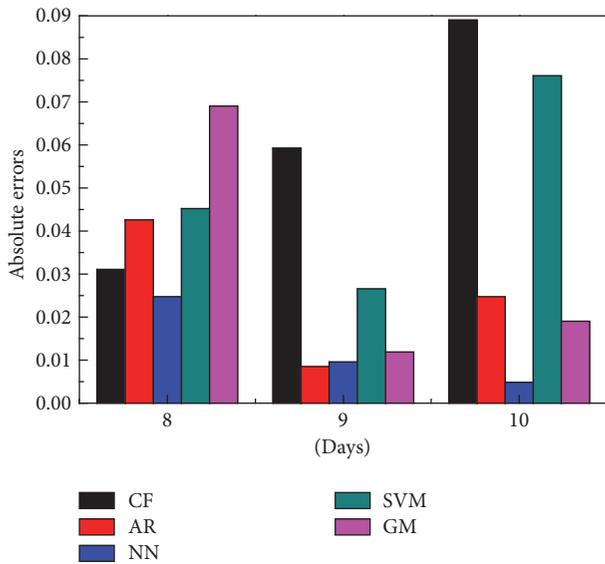


FIGURE 6: The absolute error between the actual and predicted values for the third pond.

there are two prediction error rates of test values exceeding 5%, the sum of the prediction errors of SVM model is slightly larger than that of GM model. The predicted value of CF model is relatively poor in the second experiment and the third experiment. But the predicted value of low DO of CF model for the next day (i.e., the eighth day) is satisfactory.

4. Conclusions

This paper presents the methods for predicting DO in aquaculture based on BP neural network. The combination of these three activation functions greatly improves the prediction accuracy and a large amount of calculation has been reduced. It requires less training and training data. It

is limited in computation and its prediction accuracy is the highest. The prediction error rate is less than 5%. Therefore, this method fully meets the needs of practical applications and is suitable for a wide range of promotion. The future work for us to do is to apply the neural network method in aquaculture. In particular, we can further use the oxygen machine to suppress the growth of anaerobic bacteria in water, prevent water deterioration, and improve the production efficiency of aquaculture.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants no. 61772147, no. 51478132, and no. 61100150), the University Innovation Team Construction Project of Guangdong Province (Grant no. 2015KCXTD014), Guangdong Province Natural Science Foundation of Major Basic Research and Cultivation Project (no. 2015A030308016), Guangzhou City Bureau of Cooperative Innovation Project (Grant no. 1201610005), and the Research Project of Guangzhou Education Bureau (Grant no. 1201620222).

References

- [1] A. M. Ahmed, "Prediction of dissolved oxygen in Surma River by biochemical oxygen demand and chemical oxygen demand using the artificial neural networks (ANNs)," *Journal of King Saud University-Engineering Sciences*, vol. 29, no. 2, pp. 151–158, 2017.
- [2] Q. Ding, H. Tai, D. Ma, D. Li, and L. Zhao, "Development of a smart dissolved oxygen sensor based on IEEE1451.2," *Sensor Letters*, vol. 9, no. 3, pp. 1049–1054, 2011.
- [3] S. Fiori, "Auto-regressive moving-average discrete-time dynamical systems and autocorrelation functions on real-valued Riemannian matrix manifolds," *Discrete and Continuous Dynamical Systems. Series B. A Journal Bridging Mathematics and Sciences*, vol. 19, no. 9, pp. 2785–2808, 2014.
- [4] C.-H. Ko and J.-K. Chen, "Grasping force based manipulation for multifingered hand-arm robot using neural networks," *Numerical Algebra, Control and Optimization*, vol. 4, no. 1, pp. 59–74, 2014.
- [5] F. Ascione, N. Bianco, C. De Stasio, G. M. Mauro, and G. P. Vanoli, "Artificial neural networks to predict energy performance and retrofit scenarios for any member of a building category: A novel approach," *Energy*, vol. 118, pp. 999–1017, 2017.
- [6] R. Pino-Mejías, A. Pérez-Fargallo, C. Rubio-Bellido, and J. A. Pulido-Arcas, "Comparison of linear regression and artificial neural networks models to predict heating and cooling energy demand, energy consumption and CO₂ emissions," *Energy*, vol. 118, pp. 24–36, 2017.
- [7] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, 2017.
- [8] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum

- vehicle models,” *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [9] F.-M. Tseng, H.-C. Yu, and G.-H. Tzeng, “Applied hybrid grey model to forecast seasonal time series,” *Technological Forecasting and Social Change*, vol. 67, no. 2-3, pp. 291–302, 2001.
- [10] Y. Lei, M. Guo, D.-D. Hu et al., “Short-term prediction of UT1-UTC by combination of the grey model and neural networks,” *Advances in Space Research*, vol. 59, no. 2, pp. 524–531, 2017.
- [11] J. P. Vert, “Support vector machine prediction of signal peptide cleavage site using a new class of kernels for strings,” *Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing*, vol. 7, no. 4, pp. 649–660, 2002.
- [12] Y.-C. Wang, Z.-X. Yang, and N.-Y. Deng, “Support vector machine prediction of enzyme function with conjoint triad feature and hierarchical context,” *BMC Systems Biology*, vol. 5, no. 1, article S6, 2011.
- [13] Z. Zhao, Y. Liu, and F. Luo, “Output feedback boundary control of an axially moving system with input saturation constraint,” *ISA Transactions*, vol. 68, pp. 22–32, 2016.
- [14] C. Yang, Z. Li, R. Cui, and B. Xu, “Neural network-based motion control of an underactuated wheeled inverted pendulum model,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, 2004.
- [15] H. Xiao, Z. Li, C. Yang et al., “Robust stabilization of a wheeled mobile robot using model predictive control based on neurodynamics optimization,” *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 505–516, 2017.

Research Article

Stability Analysis of Impulsive Stochastic Reaction-Diffusion Cellular Neural Network with Distributed Delay via Fixed Point Theory

Ruofeng Rao¹ and Shouming Zhong²

¹Department of Mathematics, Chengdu Normal University, Chengdu 61130, China

²College of Mathematics, University of Electronic Science and Technology of China, Chengdu 611731, China

Correspondence should be addressed to Ruofeng Rao; ruofengrao@163.com

Received 14 July 2017; Accepted 22 August 2017; Published 25 September 2017

Academic Editor: Chenguang Yang

Copyright © 2017 Ruofeng Rao and Shouming Zhong. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper investigates the stochastically exponential stability of reaction-diffusion impulsive stochastic cellular neural networks (CNN). The reaction-diffusion pulse stochastic system model characterizes the complexity of practical engineering and brings about mathematical difficulties, too. However, the difficulties have been overcome by constructing a new contraction mapping and an appropriate distance on a product space which is guaranteed to be a complete space. This is the first time to employ the fixed point theorem to derive the stability criterion of reaction-diffusion impulsive stochastic CNN with distributed time delays. Finally, an example is provided to illustrate the effectiveness of the proposed methods.

1. Introduction

In 1988, cellular neural networks (CNN) were originally introduced in [1, 2]. Since then, dynamic neural networks have received extensive attention due to their classification, associative memory, and parallel computing tasks and the ability to solve complex optimization problems. It is generally known that almost all neural networks have similar applications ([3–12]), but the key to the success of these applications lies in the stability of the system. In fact, there are a number of literatures involved in the stability analysis of CNN ([5, 7, 12–14]). In practical engineering, time delay and pulse are unavoidable. Since neural networks usually have spatial properties, due to the existence of parallel paths of various axonal sizes and lengths, it is necessary to introduce continuous distributed delays to simulate them over a given time horizon. Besides, many evolutionary processes, especially the biological neural network in biological systems and bursting rhythm models in pathology, frequency-modulated signal processing systems, are characterized by abrupt changes of states at certain time instants. In addition, electrons have diffusion behavior in inhomogeneous media.

Noise disturbance is unavoidable in real nervous systems, which is a major source of instability and poor performance in neural networks. A neural network can be stabilized or destabilized by certain stochastic inputs. The synaptic transmission in real neural networks can be viewed as a noisy process introduced by random fluctuations from the release of neurotransmitters and other probabilistic causes. Hence, the above influential factors should be also taken into consideration in stability analysis of neural networks. So, in this paper, we consider a class of impulsive stochastic reaction-diffusion cellular neural networks with distributed delay. Lyapunov function method is one of the common techniques to solve the stability of neural networks in recent decades. However, every method has its limit. Different methods lead to different criteria for stability criteria which may imply innovations. Fixed point theory and method is one of the alternative methods ([15–22]). Unlike the known literature, we try to employ Banach fixed point theory in this paper to derive the stability of impulsive stochastic reaction-diffusion cellular neural networks with distributed delay. In the next sections, we shall give some model descriptions and preliminaries and employ Banach fixed point theorem,

Hölder inequality, Burkholder-Davis-Gundy inequality, and the continuous semigroup of Laplace operators to derive the stochastically exponential stability criterion of the complex system. Of course, to overcome the difficulty of the complex mathematical model, we need to formulate a new contraction mapping on a product space. Moreover, in order to guarantee the completeness of product space, we need to give a reasonable definition of distance. Finally, an example is provided to illustrate the effectiveness of the proposed result.

2. Model Description and Preliminaries

Consider the following reaction-diffusion impulsive stochastic cellular neural networks under Dirichlet boundary value:

$$\begin{aligned} du_i(t, x) = & -q_i \operatorname{div} \nabla u_i(t, x) dt - \left[a_i u_i(t, x) \right. \\ & - \sum_{j=1}^n b_{ij} f_j(u_j(t, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(t - \tau(t), x)) \\ & \left. - \sum_{j=1}^n h_{ij} \int_{t-\rho(t)}^t f_j(u_j(s, x)) ds \right] dt \\ & + \sigma_i(u_i(t, x)) dw_i(t), \end{aligned} \quad (1)$$

$$t \neq t_k, \quad x \in Y, \quad i \in \mathcal{N}$$

$$u(t_k^+, x) = u(t_k^-, x) + g(u(t_k, x)),$$

$$x \in Y, \quad k = 1, 2, \dots$$

$$u_i(t, x) = \zeta_i(t, x), \quad \forall (s, x) \in [-\tau, 0] \times Y$$

$$u(t, x) = 0, \quad \forall (t, x) \in [0, +\infty) \times \partial Y,$$

where $Y \subset R^m$ is a bounded domain with the smooth boundary ∂Y . $u_i(t, x)$ is the state variable of the i th neuron at time t and in space variable x for $i \in \mathcal{N}$ with $\mathcal{N} \triangleq \{1, 2, \dots, n\}$. f_i denotes the active function of neuron. a_i is the rate with which the i th neuron will reset its potential to the resting state in isolation when disconnected from the networks and the external inputs. b_{ij} , c_{ij} , and h_{ij} are elements of feedback template. Let $\{w_i(t), t \geq 0\}$ be a real-valued Brownian motion defined on the complete probability space $\{\Omega, \mathcal{F}, \mathbb{P}\}$ which has natural filtration $\{\mathcal{F}_t\}_{t \geq 0}$. Denote by $\mathcal{L}^2(Y)$ the space of all real-valued square integrable functions with the inner product $\langle \xi, \eta \rangle = \int_Y \xi(x) \eta(x) dx$, for $\xi, \eta \in \mathcal{L}^2(Y)$ which derives the norm $\|\xi\| = (\int_Y \xi^2(x) dx)^{1/2}$ for $\xi \in \mathcal{L}^2(Y)$. $\sigma_i(\cdot)$ is a Borel measurable function. Denote by $\Delta = \sum_{j=1}^m (\partial^2 / \partial x_j^2)$ the Laplace operator, with domain $\mathcal{D}(\Delta) = W_0^{1,2}(Y) \cap W_0^{2,2}(Y)$, which generates a strongly continuous semigroup $e^{-q_i t \Delta}$, where $W_0^{1,2}(Y)$ and $W_0^{2,2}(Y)$ are the Sobolev spaces with compactly supported sets. $\operatorname{div} \nabla u_i(t, x)$ denotes the divergence of $\nabla u_i(t, x)$ (see, e.g., [25, 26]). q_i is the diffusion coefficient, and time delays $\tau(t), \rho(t) \in [0, \tau]$. Besides, initial value $\zeta_i(t, x)$ is continuous for $(s, x) \in [-\tau, 0] \times Y$. The fixed impulsive moments t_k ($k = 1, 2, \dots$) satisfy

$0 < t_1 < t_2 < \dots$ with $\lim_{k \rightarrow +\infty} t_k = +\infty$. $u(t_k^+, x)$ and $u(t_k^-, x)$ stand for the right-hand and left-hand limit of $u(t, x)$ at time t_k , respectively. Further, suppose that $u(t_k^-, x) = \lim_{t \rightarrow t_k^-} u(t, x) = u(t_k, x)$, $k = 1, 2, \dots$

In this paper, we assume that

(H1) $\|e^{-q_i t \Delta}\| \leq M e^{-\gamma t}$, where $M > 0$ and $\gamma > 0$ are constants;

(H2) f_i, g_i , and σ_i are Lipschitz continuous with Lipschitz constants $L_i > 0, G_i$, and $T_i > 0$ for $i \in \mathcal{N}$, respectively. In addition, $f_i(0) = g_i(0) = 0 = \sigma_i(0), \forall i \in \mathcal{N}$.

Definition 1. For any $T > 0$ and $x \in Y$, a stochastic process $u = \{(u_1(t, x), u_2(t, x), \dots, u_n(t, x))\}_{[0, T]}$ is called a mild solution of impulsive system (1) if, for any $i \in \mathcal{N}$, $u_i(t, x) \in \mathcal{C}([0, T]; \mathcal{L}^2(Y))$ and, for any $t \in [0, T]$, $u_i(t, x)$ is adapted to \mathcal{F}_t with

$$\mathbb{P} \left\{ \omega : \int_0^t \int_Y |u_i(s)|^2 dx ds < \infty \right\} = 1, \quad (2)$$

and the following stochastic integral equations hold for all $i \in \mathcal{N}$, a.s. for any $t \in [0, T]$ and $x \in Y$,

$$\begin{aligned} u_i(t, x) = & e^{-q_i t \Delta} \zeta(0, x) - \int_0^t e^{-q_i(t-\theta) \Delta} \left[a_i u_i(\theta, x) \right. \\ & - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \\ & \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \\ & + \int_0^t e^{-q_i(t-\theta) \Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \\ & + e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)), \quad t \geq 0 \end{aligned} \quad (3)$$

$$u_i(t, x) = \zeta_i(t, x), \quad \forall (s, x) \in [-\tau, 0] \times Y,$$

$$u(t, x) = 0, \quad \forall (t, x) \in [0, +\infty) \times \partial Y.$$

Remark 2. In Definition 1, the mild solution of impulsive system (1) is well defined due to [24, Lemma 3.1].

Lemma 3 (Hölder inequality). *Assume that $1/p + 1/q = 1$ with $p > 1$, and $\varphi(x) \in \mathcal{L}^p(Y)$, $\phi \in \mathcal{L}^q(Y)$; then,*

$$\begin{aligned} & \int_Y \varphi(x) \phi(x) dx \\ & \leq \left(\int_Y |\varphi(x)|^p dx \right)^{1/p} \left(\int_Y |\phi(x)|^q dx \right)^{1/q}. \end{aligned} \quad (4)$$

Lemma 4 (Banach contraction mapping principle). *Let Θ be a contraction operator on a complete metric space Γ ; then there exists a unique point $u \in \Gamma$ for which $\Theta(u) = u$.*

3. Main Result: Stochastically Exponential Stability

Theorem 5. Assume that (H1) and (H2) hold. Then, CNN (1) is stochastically exponentially mean square stable if the following condition holds:

$$0 < \kappa < 1, \quad (5)$$

where $\mu = \inf_{k=1,2,\dots} (t_{k+1} - t_k) > 0$ and

$$\begin{aligned} \kappa \triangleq & 6M^2 \left[\frac{1}{\gamma^2} \left(\max_{i \in \mathcal{N}} a_i^2 \right) \right. \\ & + n \frac{1}{\gamma^2} \max_{i \in \mathcal{N}} \left(\sum_{j=1}^n (|b_{ij}|^2 + |c_{ij}|^2) L_j^2 \right) + \frac{n\tau^2}{\gamma^2} \\ & \left. + 2M^2 \left(1 + \frac{1}{\gamma^2 \mu^2} \right) \left(\max_{i \in \mathcal{N}} G_i^2 \right) + \frac{2}{\gamma} \left(\max_{i \in \mathcal{N}} T_i^2 \right) \right]. \end{aligned} \quad (6)$$

Proof. Firstly, we need to formulate a contraction mapping on a product space.

Let Γ_i be the Banach space of all \mathcal{F}_t -adapted mean square continuous processes consisting of functions $u_i(t, x)$ at $t \geq 0$ with $t \neq t_k$ such that $\mathbb{E}(e^{\alpha t} \|u_i(t, x)\|^2) \rightarrow 0$ as $t \rightarrow +\infty$, where $\alpha \in (0, \gamma)$ is a positive scalar. Now, we construct an operator $\Theta \triangleq (\Theta_1, \Theta_2, \dots, \Theta_i, \dots, \Theta_n)$ with $\Theta_i : \Gamma_i \rightarrow \Gamma_i$ as follows:

$$\begin{aligned} \Theta_i(u_i)(t, x) = & e^{-q_i t \Delta} \zeta(0, x) - \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) \right. \\ & - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \\ & \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \\ & + \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \\ & + e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)), \quad t \geq 0, \end{aligned} \quad (7)$$

$$\Theta_i(u_i)(t, x) = \zeta_i(t, x), \quad (s, x) \in [-\tau, 0] \times Y \quad (8)$$

$$\Theta_i(u_i)(t, x) = 0, \quad \forall (t, x) \in [0, +\infty) \times \partial Y.$$

Equipped with the following distance:

$$\text{dist}(u, v) = \left(\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) - v_i(t, x)\|^2 \right)^{1/2}, \quad (9)$$

$$\forall u, v \in \Gamma_1 \times \Gamma_2 \times \dots \times \Gamma_n,$$

$\Gamma_1 \times \Gamma_2 \times \dots \times \Gamma_n$ becomes a complete metric space, where $u = u(t, x) = (u_1(t, x), u_2(t, x), \dots, u_n(t, x))^T$, $v = v(t, x) = (v_1(t, x), v_2(t, x), \dots, v_n(t, x))^T$.

Next, we are to apply contractive mapping theory to complete the proof via three steps.

Step 1. From (7), for $t \in [0, +\infty) \setminus \{t_k\}_{k=1}^{\infty}$, we claim that $\Theta_i(u_i)(t)$ is mean square continuous. Indeed, let δ be a small enough scalar:

$$\begin{aligned} & \mathbb{E} \left\| \Theta_i(u_i)(t + \delta, x) - \Theta_i(u_i)(t, x) \right\|^2 \\ & \leq 4\mathbb{E} \left\| e^{-q_i(t+\delta)\Delta} \zeta(0, x) - e^{-q_i t \Delta} \zeta(0, x) \right\|^2 \\ & \quad + 4\mathbb{E} \left\| \int_0^{t+\delta} e^{-q_i(t+\delta-\theta)\Delta} \left[a_i u_i(\theta, x) \right. \right. \\ & \quad - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \\ & \quad \left. \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \right. \\ & \quad - \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \\ & \quad \left. - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right. \\ & \quad \left. \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \right\|^2 \\ & \quad + 4\mathbb{E} \left\| \int_0^{t+\delta} e^{-q_i(t+\delta-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right. \\ & \quad \left. - \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right\|^2 \\ & \quad + 4\mathbb{E} \left\| e^{-q_i(t+\delta)\Delta} \sum_{0 < t_k < t+\delta} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right. \\ & \quad \left. - e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2. \end{aligned} \quad (10)$$

Firstly, we estimate

$$\begin{aligned} & \mathbb{E} \left\| e^{-q_i(t+\delta)\Delta} \zeta(0, x) - e^{-q_i t \Delta} \zeta(0, x) \right\|^2 \\ & \leq \mathbb{E} \left\| (e^{-q_i \delta \Delta} - 1) e^{-q_i t \Delta} \zeta(0, x) \right\|^2 \rightarrow 0, \end{aligned} \quad (11)$$

if $\delta \rightarrow 0$.

Next, we evaluate

$$\begin{aligned} & \mathbb{E} \left\| \int_0^{t+\delta} e^{-q_i(t+\delta-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \right. \\ & \quad \left. \left. - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right] d\theta \right. \\ & \quad \left. - \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \right. \\ & \quad \left. \left. - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right] d\theta \right\|^2 \end{aligned}$$

$$\begin{aligned}
& - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \Big] d\theta \\
& - \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \\
& \left. - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right. \\
& \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \Big\|^2 \\
& \leq 2\mathbb{E} \left\| \int_t^{t+\delta} e^{-q_i(t+\delta-\theta)\Delta} \left[a_i u_i(\theta, x) \right. \right. \\
& \left. \left. - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right. \right. \\
& \left. \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \right\|^2 \\
& + 2\mathbb{E} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \right. \\
& \left. \left. - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \right. \right. \\
& \left. \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] (e^{-q_i\delta\Delta} - 1) d\theta \right\|^2 \\
& \longrightarrow 0, \quad \text{if } \delta \longrightarrow 0.
\end{aligned} \tag{12}$$

Via Burkholder-Davis-Gundy inequality, we can conclude that if $\delta \rightarrow 0$,

$$\begin{aligned}
& \mathbb{E} \left\| \int_0^{t+\delta} e^{-q_i(t+\delta-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right. \\
& \left. - \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right\|^2 \\
& \leq 8\mathbb{E} \int_0^t M^2 e^{-2\gamma(t-\theta)} \|\sigma_i(u_i(\theta, x)) (e^{-q_i\delta\Delta} - 1)\|^2 d\theta \\
& + 8\mathbb{E} \int_t^{t+\delta} M^2 e^{-2\gamma(t+\delta-\theta)} \|\sigma_i(u_i(\theta, x))\|^2 d\theta \longrightarrow 0.
\end{aligned} \tag{13}$$

Due to $t \neq t_k$, it is obvious that

$$\begin{aligned}
& \mathbb{E} \left\| e^{-q_i(t+\delta)\Delta} \sum_{0 < t_k < t+\delta} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right. \\
& \left. - e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \longrightarrow 0, \\
& \text{if } \delta \longrightarrow 0.
\end{aligned} \tag{14}$$

So, we have proved from (10)–(14) that $\Theta_i(u_i)(t)$ is mean square continuous at $t \geq 0$ with $t \neq t_k$.

Next, we claim that

$$\begin{aligned}
& \lim_{\delta \rightarrow 0^+} \Theta_i(u_i)(t_k + \delta) = \Theta_i(u_i)(t_k) + g(u_i(t_k)), \\
& \lim_{\delta \rightarrow 0^-} \Theta_i(u_i)(t_k + \delta) = \Theta_i(u_i)(t_k).
\end{aligned} \tag{15}$$

Indeed, obviously, (11)–(13) hold for all $t = t_k$, too. In addition, let $\delta > 0$ be small enough:

$$\begin{aligned}
& e^{-q_i(t_k+\delta)\Delta} \sum_{0 < t_j < t_k+\delta} e^{q_i t_j \Delta} g_i(u_i(t_j, x)) \\
& - e^{-q_i t_k \Delta} \sum_{0 < t_j < t_k} e^{q_i t_j \Delta} g_i(u_i(t_j, x)) \\
& = g_i(u(t_k, x)), \quad \delta \longrightarrow 0^+.
\end{aligned} \tag{16}$$

On the other hand, let $\delta < 0$ be small enough:

$$\begin{aligned}
& e^{-q_i(t_k+\delta)\Delta} \sum_{0 < t_j < t_k+\delta} e^{q_i t_j \Delta} g_i(u_i(t_j, x)) \\
& - e^{-q_i t_k \Delta} \sum_{0 < t_j < t_k} e^{q_i t_j \Delta} g_i(u_i(t_j, x)) = 0,
\end{aligned} \tag{17}$$

$\delta \longrightarrow 0^-$.

This together with (16) implies that (15) holds.

Step 2. We claim that

$$\mathbb{E} (e^{\alpha t} \|\Theta_i(u_i(t, x))\|^2) \longrightarrow 0, \quad \text{if } t \longrightarrow +\infty. \tag{18}$$

Indeed, we have the following inequality similar to (10):

$$\begin{aligned}
& \mathbb{E} (e^{\alpha t} \|\Theta_i(u_i)(t, x)\|^2) \leq 4\mathbb{E} (e^{\alpha t} \|e^{-q_i t \Delta} \zeta(0, x)\|^2) \\
& + 4\mathbb{E} \left\{ e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) \right. \right. \right.
\end{aligned}$$

$$\begin{aligned}
& - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \\
& - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \Big] d\theta \Big\| \Big\|^2 \\
& + 4\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right\|^2 \right) \\
& + 4\mathbb{E} \left(e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right), \\
& t \geq 0. \tag{19}
\end{aligned}$$

Condition (H1) yields

$$\begin{aligned}
\mathbb{E} \left(e^{\alpha t} \left\| e^{-q_i t \Delta} \zeta(0, x) \right\|^2 \right) & \leq \mathbb{E} \left(M^2 e^{-(2\gamma-\alpha)t} \left\| \zeta(0, x) \right\|^2 \right) \\
& \rightarrow 0, \quad \text{if } t \rightarrow +\infty. \tag{20}
\end{aligned}$$

For any given $\varepsilon > 0$, the assumption $\mathbb{E}(e^{\alpha t} \|u_i(t, x)\|^2) \rightarrow 0$ tells us that there exists $t_* > 0$ such that

$$\mathbb{E} \left(e^{\alpha t} \|u_i(t, x)\|^2 \right) < \varepsilon, \quad \forall t \geq t_*. \tag{21}$$

Moreover, Hölder inequality gives

$$\begin{aligned}
\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} a_i u_i(\theta, x) d\theta \right\|^2 \right) & \leq \frac{M^2 a_i^2}{\gamma} \\
\cdot \mathbb{E} \left(e^{\alpha t} \int_0^t e^{-\gamma(t-\theta)} \|u_i(\theta, x)\|^2 d\theta \right) & \leq \frac{M^2 a_i^2}{\gamma} \\
\cdot \mathbb{E} \left(e^{-(\gamma-\alpha)t} t^* e^{\gamma t^*} \max_{\theta \in [0, t_*]} (\|u_i(\theta, x)\|^2) + \varepsilon \frac{1}{\gamma - \alpha} \right), \tag{22}
\end{aligned}$$

which together with the arbitrariness of ε derives

$$\begin{aligned}
\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} a_i u_i(\theta, x) d\theta \right\|^2 \right) & \rightarrow 0, \\
& \text{if } t \rightarrow +\infty. \tag{23}
\end{aligned}$$

Besides,

$$\begin{aligned}
& \mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) d\theta \right\|^2 \right) \\
& \leq \mathbb{E} \left(M \sum_{j=1}^n |b_{ij}| L_j e^{-(\gamma-\alpha)t} \int_0^t e^{\gamma\theta} \|u_j(\theta, x)\| d\theta \right)^2. \tag{24}
\end{aligned}$$

Using similar methods of (21) and (22), we can deduce from (24) that

$$\begin{aligned}
\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) d\theta \right\|^2 \right) & \rightarrow 0, \\
& \text{if } t \rightarrow +\infty. \tag{25}
\end{aligned}$$

Similar to that of (24) and (22), we can also obtain

$$\begin{aligned}
& \mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) d\theta \right\|^2 \right) \\
& \leq \mathbb{E} \left[M \sum_{j=1}^n |c_{ij}| L_j \frac{1}{\gamma} \left(e^{-(\gamma-\alpha)t} e^{\gamma\tau} (t_* + \tau) \right. \right. \\
& \cdot \max_{s \in [-\tau, t_* + \tau]} (e^{\gamma s} \|u_j(s, x)\|^2) + \varepsilon e^{\gamma\tau} \frac{1}{\gamma - \alpha} \Big) \Big]. \tag{26}
\end{aligned}$$

Now, similar to that of (22), we know from (26) that

$$\begin{aligned}
& \mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) d\theta \right\|^2 \right) \\
& \rightarrow 0, \quad \text{if } t \rightarrow +\infty. \tag{27}
\end{aligned}$$

Similarly, Hölder inequality yields

$$\begin{aligned}
& \mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds d\theta \right\|^2 \right) \\
& \leq M^2 \mathbb{E} \left[\sum_{j=1}^n |h_{ij}| L_j \frac{\tau}{\gamma} \right. \\
& \cdot \left(e^{-(\gamma-\alpha)t} \tau \max_{\theta \in [-\tau, t_* + \tau]} \|u_j(s, x)\|^2 \frac{1}{\gamma} e^{\gamma(t_* + \tau)} + \varepsilon \tau e^{\alpha\tau} \right. \\
& \cdot \left. \left. \frac{1}{\gamma - \alpha} \right) \right]. \tag{28}
\end{aligned}$$

Similar to (22), we can conclude from (28) that

$$\begin{aligned}
& \mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds d\theta \right\|^2 \right) \\
& \rightarrow 0, \quad \text{if } t \rightarrow +\infty. \tag{29}
\end{aligned}$$

Hence,

$$\begin{aligned}
& \mathbb{E} \left\{ e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \left[a_i u_i(\theta, x) - \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) \right. \right. \right. \\
& - \sum_{j=1}^n c_{ij} f_j(u_j(\theta - \tau(\theta), x)) \\
& \left. \left. \left. - \sum_{j=1}^n h_{ij} \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds \right] d\theta \right\|^2 \right\}
\end{aligned}$$

$$\begin{aligned}
&\leq 4\mathbb{E} \left\| e^{\alpha t} \int_0^t e^{-q_i(t-\theta)\Delta} a_i u_i(\theta, x) d\theta \right\|^2 \\
&+ 4\mathbb{E} \left\| e^{\alpha t} \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n b_{ij} f_j(u_j(\theta, x)) d\theta \right\|^2 \\
&+ 4\mathbb{E} \left\| e^{\alpha t} \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n c_{ij} f_j(u_j(\theta, x)) \right. \\
&\left. - \tau(\theta, x) d\theta \right\|^2 + 4\mathbb{E} \left\| e^{\alpha t} \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n h_{ij} \right. \\
&\left. \cdot \int_{\theta-\rho(\theta)}^{\theta} f_j(u_j(s, x)) ds d\theta \right\|^2 \rightarrow 0, \\
&\text{if } t \rightarrow +\infty. \tag{30}
\end{aligned}$$

Burkholder-Davis-Gundy inequality and Hölder inequality derive

$$\begin{aligned}
&\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right\|^2 \right) \\
&\leq 8\mathbb{E} \left(M^2 T_i^2 e^{-(2\gamma-\alpha)t} \max_{\theta \in [0, t^*]} e^{2\gamma\theta} \|u_i(\theta, x)\|^2 \right) \tag{31} \\
&+ 8\varepsilon \mathbb{E} \left(M^2 T_i^2 \sqrt{\frac{1}{4\gamma(\gamma-\alpha)}} \right),
\end{aligned}$$

which together with the arbitrariness of ε implies

$$\mathbb{E} \left(e^{\alpha t} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sigma_i(u_i(\theta, x)) dw_i(\theta) \right\|^2 \right) \rightarrow 0, \tag{32}$$

if $t \rightarrow +\infty$.

Next, we may assume that $t_{l-1} < t_* \leq t_l$ and $t_{j-1} < t \leq t_j$.

In addition, one can deduce from (H1)

$$\begin{aligned}
&\mathbb{E} \left\{ e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{0 < t_k \leq t_*} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right\} \\
&\leq \mathbb{E} \left[M e^{(\alpha/2)t} e^{-\gamma t} \left(\sum_{0 < t_k \leq t_l} M e^{\gamma t_k} G_i \|u_i(t_k, x)\| \right) \right]^2 \tag{33} \\
&\rightarrow 0, \text{ if } t \rightarrow +\infty.
\end{aligned}$$

Besides, we can estimate by means of definite integral

$$\begin{aligned}
&\mathbb{E} \left\{ e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{t_* < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right\} \\
&\leq \varepsilon G_i^2 M^4 \mathbb{E} \left(e^{-(1/2)(2\gamma-\alpha)t} \sum_{t_l \leq t_k \leq t_{j-1}} e^{(1/2)(2\gamma-\alpha)t_k} \right)^2 \tag{34} \\
&\leq \varepsilon \mathbb{E} \left(M^2 G_i \left(1 + \frac{2}{\mu(2\gamma-\alpha)} \right) \right)^2.
\end{aligned}$$

Moreover, the arbitrariness of ε implies

$$\mathbb{E} \left\{ e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{t_* < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right\} \rightarrow 0, \tag{35}$$

if $t \rightarrow +\infty$.

Hence, if $t \rightarrow +\infty$,

$$\begin{aligned}
&\mathbb{E} \left\{ e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right\} \\
&\leq 2\mathbb{E} \left(e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{0 < t_k \leq t_*} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right) \tag{36} \\
&+ 2\mathbb{E} \left(e^{\alpha t} \left\| e^{-q_i t \Delta} \sum_{t_* < t_k < t} e^{q_i t_k \Delta} g_i(u_i(t_k, x)) \right\|^2 \right) \\
&\rightarrow 0.
\end{aligned}$$

Combining (19), (20), (23), (30), (32), and (36) results in (18).

Step 3. Finally, we claim that Θ is a contractive mapping on $\Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$.

Indeed, from the above two steps, we know $\Theta_i(\Gamma_i) \subset \Gamma_i$, and then $\Theta(\Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n) \subset \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$.

On the other hand, for any $i \in \mathcal{N}$ and $u, v \in \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$,

$$\begin{aligned}
&\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|\Theta_i(u_i)(t, x) - \Theta_i(v_i)(t, x)\|^2 \\
&\leq 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} a_i(u_i(\theta, x)) \right. \\
&\left. - v_i(\theta, x) d\theta \right\|^2 + 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \right. \\
&\left. \cdot \sum_{j=1}^n b_{ij} (f_j(u_j(\theta, x)) - f_j(v_j(\theta, x))) d\theta \right\|^2
\end{aligned}$$

$$\begin{aligned}
& + 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n c_{ij} \right. \\
& \cdot \left(f_j(u_j(\theta - \tau(\theta), x)) \right. \\
& \left. - f_j(v_j(\theta - \tau(\theta), x)) \right) d\theta \left\| ^2 \\
& + 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n h_{ij} \right. \\
& \cdot \left(\int_{\theta-\rho(\theta)}^{\theta} [f_j(u_j(s, x)) - f_j(v_j(s, x))] ds \right) d\theta \left\| ^2 \\
& + 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} [\sigma_i(u_i(\theta, x)) \right. \right. \\
& \left. \left. - \sigma_i(v_i(\theta, x))] dw_i(\theta) \right\| ^2 \\
& + 6\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} [g_i(u_i(t_k, x)) \right. \right. \\
& \left. \left. - g_i(v_i(t_k, x))] \right\| ^2.
\end{aligned} \tag{37}$$

Besides, it follows by the Hölder inequality that

$$\begin{aligned}
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} a_i(u_i(\theta, x) - v_i(\theta, x)) d\theta \right\| ^2 \leq M^2 \\
& \cdot \frac{1}{\gamma^2} \left(\max_{i \in \mathcal{N}} a_i^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(\theta, x) - v_i(\theta, x)\|^2, \\
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n b_{ij} \right. \\
& \cdot \left(f_j(u_j(\theta, x)) - f_j(v_j(\theta, x)) \right) d\theta \left\| ^2 \leq nM^2 \frac{1}{\gamma^2} \\
& \cdot \max_{i \in \mathcal{N}} \left(\sum_{j=1}^n |b_{ij}|^2 L_j^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) - v_i(t, x)\|^2, \\
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \sum_{j=1}^n c_{ij} \right. \\
& \cdot \left(f_j(u_j(\theta - \tau(\theta), x)) - f_j(v_j(\theta - \tau(\theta), x)) \right) d\theta \left\| ^2 \\
& \leq nM^2 \frac{1}{\gamma^2} \max_{i \in \mathcal{N}} \left(\sum_{j=1}^n |c_{ij}|^2 L_j^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) \right. \\
& \left. - v_i(t, x)\|^2,
\end{aligned}$$

$$\begin{aligned}
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \right. \\
& \cdot \sum_{j=1}^n h_{ij} \left(\int_{\theta-\rho(\theta)}^{\theta} [f_j(u_j(s, x)) - f_j(v_j(s, x))] ds \right) d\theta \left\| ^2 \\
& \leq nM^2 \tau^2 \frac{1}{\gamma^2} \max_{i \in \mathcal{N}} \left(\sum_{j=1}^n |h_{ij}|^2 L_j^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) \right. \\
& \left. - v_i(t, x)\|^2, \\
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| e^{-q_i t \Delta} \sum_{0 < t_k < t} e^{q_i t_k \Delta} \right. \\
& \cdot [g_i(u_i(t_k, x)) - g_i(v_i(t_k, x))] \left\| ^2 \leq M^4 \left(\max_{i \in \mathcal{N}} G_i^2 \right) \\
& \cdot \left[2\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) - v_i(t, x)\|^2 \right. \\
& \left. + 2\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left(\frac{1}{\mu} e^{-\gamma t} \int_0^t e^{\gamma s} \|u_i(s, x) - v_i(s, x)\| ds \right)^2 \right] \\
& \leq 2M^4 \left(1 + \frac{1}{\gamma^2 \mu^2} \right) \left(\max_{i \in \mathcal{N}} G_i^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(t, x) \right. \\
& \left. - v_i(t, x)\|^2,
\end{aligned} \tag{38}$$

where we assume that $t_{j-1} < t \leq t_j$.

In addition, it follows from Burkholder-Davis-Gundy inequality that

$$\begin{aligned}
& \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \left\| \int_0^t e^{-q_i(t-\theta)\Delta} \right. \\
& \cdot [\sigma_i(u_i(\theta, x)) - \sigma_i(v_i(\theta, x))] dw_i(\theta) \left\| ^2 \leq \frac{2}{\gamma} \\
& \cdot M^2 \left(\max_{i \in \mathcal{N}} T_i^2 \right) \mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} \|u_i(\theta, x) - v_i(\theta, x)\|^2.
\end{aligned} \tag{39}$$

Now, combining (37)–(39) gives

$$\begin{aligned}
& \text{dist}(\Theta(u), \Theta(v)) \leq \sqrt{\kappa} \text{dist}(u, v), \\
& \forall u, v \in \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n,
\end{aligned} \tag{40}$$

where κ is defined as (6), satisfying $0 < \kappa < 1$. This implies that $\Theta : \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n \rightarrow \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$ is a contraction mapping such that there exists the fixed point $u \triangleq (u_1(t, x), u_2(t, x), \dots, u_n(t, x))$ of Θ in $\Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$, which implies that u is a solution of CNN (1), satisfying $\mathbb{E}(e^{\alpha t} \|\Theta_i(u_i(t, x))\|^2) \rightarrow 0, t \rightarrow +\infty$ so that $\mathbb{E} \max_{i \in \mathcal{N}} \sup_{t \geq -\tau} (e^{\alpha t} \|\Theta_i(u_i(t, x))\|^2) \rightarrow 0, t \rightarrow +\infty$. Therefore, CNN (1) is stochastically exponentially mean square stable. \square

TABLE 1: Comparison of the complexity of system models in the literature related to fixed point theory.

	Theorem 5	[16]	[15]	[23]	[24]
Using fixed point theory	Yes	Yes	Yes	Yes	Yes
Impulse model	Yes	No	No	Yes	Yes
Distributed delays	Yes	No	No	No	No
Reaction-diffusion model	Yes	No	No	No	No
Itô stochastic model	Yes	No	No	No	No
Equations type	Integrodifferential (partial) eq.	Differential eq.	Differential eq.	Differential eq.	Integrodifferential eq.
Stability type	Stochastically exponential	Exponential	Exponential	Exponential	Exponential

4. Numerical Example

Consider the following impulsive stochastic reaction-diffusion CNN with distributed delay:

$$\begin{aligned}
du_i(t, x) = & -q_i \operatorname{div} \nabla u_i(t, x) dt - \left[a_i u_i(t, x) \right. \\
& - \sum_{j=1}^n b_{ij} \sin\left(\frac{j}{10} u_j(t, x)\right) \\
& - \sum_{j=1}^n c_{ij} \sin\frac{j}{10}\left(u_j(t - \tau(t), x)\right) \\
& \left. - \sum_{j=1}^n h_{ij} \int_{t-\rho(t)}^t \sin\left(\frac{j}{10} u_j(s, x)\right) ds \right] dt \\
& + \sin(0.05 i u_i(t, x)) dw_i(t),
\end{aligned} \tag{41}$$

$$t \neq t_k, \quad x \in Y, \quad i \in \mathcal{N}$$

$$\begin{aligned}
u(t_k^+, x) = & u(t_k^-, x) + 0.1 j \sin(u(t_k, x)), \\
& x \in Y, \quad k = 1, 2, \dots
\end{aligned}$$

$$u_i(t, x) = \zeta_i(t, x), \quad (s, x) \in [-\tau, 0] \times Y$$

$$u(t, x) = 0, \quad u \in [0, +\infty) \times \partial Y,$$

where we suppose $Y = (0, \pi)$, $n = 2$, $\tau = 3$, $\mu = 1.5$, $a_i = 0.5i$, $b_{ij} = 0.01(i + j) = c_{ij} = h_{ij}$, and $q_i = -1$. Then, via computing the eigenfunctions of $-\Delta$, we can obtain that $\|e^{t\Delta}\| \leq e^{-\pi^2 t}$, $t \geq 0$, so that we can take $\gamma = \pi^2$, $M = 1$. In addition, differential mean value theorem yields

$$\begin{aligned}
& \left| \sin\left(\frac{j}{10} u_j(t, x)\right) - \sin\left(\frac{j}{10} v_j(t, x)\right) \right| \\
& \leq \frac{j}{10} |u_j(t, x) - v_j(t, x)|,
\end{aligned} \tag{42}$$

and then we get $L_j = j/10$, $j = 1, 2$. Similarly, we can compute that $G_i = 0.1i$, $T_i = 0.05i$, and $i = 1, 2$. Finally, we can compute (6) on a computer running Matlab software, obtaining $\kappa = 0.8716 \in (0, 1)$. Therefore, Theorem 5 tells us that CNN (41) is stochastically exponentially mean square stable.

Table 1 is presented to compare the complexity of neural networks investigated in various literatures via fixed point theorems and techniques.

Remark 6. Impulsive reaction-diffusion Itô-type stochastic model gives a lot of mathematical difficulties in deriving the stability criterion. Motivated by some methods and techniques of the above-mentioned literature ([3–31]), this is the first time for us to analyze such a complex model by way of fixed point theorem. Our model is closer to real engineering so that it is more complex than those of the previous literature, and we utilize Banach fixed point theorem, Hölder inequality, Burkholder-Davis-Gundy inequality, and the continuous semigroup of Laplace operators to overcome the difficulties. Besides, the distance defined in this paper satisfies the triangle inequality, which is another point different from those of previous related literatures.

5. Conclusions

Since our CNN model involves pulse and Laplacian operators, our model is different from the previous model ([15–22]), which also implies some difficulties in mathematical techniques. Motivated by the previous literature related to fixed point theory ([15–22, 25–31]), the authors employed Banach fixed point theorem, Hölder inequality, Burkholder-Davis-Gundy inequality, and the continuous semigroup of Laplace operators to derive the stochastically exponential stability criterion of impulsive stochastic reaction-diffusion cellular neural networks with distributed delay.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Basic Research Program of China (2010CB732501), Scientific Research Fund of Science Technology Department of Sichuan Province (2010JY0057, 2012JY010), Sichuan Educational Committee Science Foundation (08ZB002, 12ZB349, 14ZA0274), and the Initial Founding of Scientific Research for Chengdu Normal University Introduction of Talents.

References

- [1] L. O. Chua and L. Yang, "Cellular neural networks: theory," *Institute of Electrical and Electronics Engineers. Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1257–1272, 1988.
- [2] L. O. Chua and L. Yang, "Cellular neural networks: applications," *Institute of Electrical and Electronics Engineers. Transactions on Circuits and Systems*, vol. 35, no. 10, pp. 1273–1290, 1988.
- [3] X. Li and R. Rakkiyappan, "Impulse controller design for exponential synchronization of chaotic neural networks with mixed delays," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 6, pp. 1515–1523, 2013.
- [4] B. Wang, J. Cheng, and J. Zhan, "A sojourn probability approach to fuzzy-model-based reliable control for switched systems with mode-dependent time-varying delays," *Nonlinear Analysis. Hybrid Systems*, vol. 26, pp. 239–253, 2017.
- [5] X. Li, R. Rakkiyappan, and P. Balasubramaniam, "Existence and global stability analysis of equilibrium of fuzzy cellular neural networks with time delay in the leakage term under impulsive perturbations," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 348, no. 2, pp. 135–155, 2011.
- [6] Q. Song, H. Yan, Z. Zhao, and Y. Liu, "Global exponential stability of impulsive complex-valued neural networks with both asynchronous time-varying and continuously distributed delays," *Neural Networks*, vol. 81, pp. 1–10, 2016.
- [7] M. Kimura, R. Morita, S. Sugisaki, T. Matsuda, and Y. Nakashima, "Cellular neural network formed by simplified processing elements composed of thin-film transistors," *Neurocomputing*, vol. 248, pp. 112–119, 2017.
- [8] J. Cheng, J. H. Park, Y. Liu, Z. Liu, and L. Tang, "Finite-time H_∞ fuzzy control of nonlinear Markovian jump delayed systems with partly uncertain transition descriptions," *Fuzzy Sets and Systems*, vol. 314, pp. 99–115, 2017.
- [9] X. Li, C. Ding, and Q. Zhu, "Synchronization of stochastic perturbed chaotic neural networks with mixed delays," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 347, no. 7, pp. 1266–1280, 2010.
- [10] B. Wang, J. Cheng, A. Al-Barakati, and H. M. Fardoun, "A mismatched membership function approach to sampled-data stabilization for T-S fuzzy systems with time-varying delayed signals," *Signal Processing*, vol. 140, pp. 161–170, 2017.
- [11] K. Shi, X. Liu, Y. Tang, H. Zhu, and S. Zhong, "Some novel approaches on state estimation of delayed neural networks," *Information Sciences*, vol. 372, pp. 313–331, 2016.
- [12] Q. Song and J. Cao, "Dynamical behaviors of discrete-time fuzzy cellular neural networks with variable delays and impulses," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 345, no. 1, pp. 39–59, 2008.
- [13] R. Jia, "Finite-time stability of a class of fuzzy cellular neural networks with multi-proportional delays," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 319, pp. 70–80, 2017.
- [14] Y. G. Kao, L. Shi, J. Xie, and H. R. Karimi, "Global exponential stability of delayed Markovian jump fuzzy cellular neural networks with generally incomplete transition probability," *Neural Networks*, vol. 63, pp. 18–30, 2015.
- [15] B. Liu, "Global exponential stability for BAM neural networks with time-varying delays in the leakage terms," *Nonlinear Analysis. Real World Applications. An International Multidisciplinary Journal*, vol. 14, no. 1, pp. 559–566, 2013.
- [16] L. Zhou, "Novel global exponential stability criteria for hybrid BAM neural networks with proportional delays," *Neurocomputing*, vol. 161, pp. 99–106, 2015.
- [17] J. Luo, "Fixed points and stability of neutral stochastic delay differential equations," *Journal of Mathematical Analysis and Applications*, vol. 334, no. 1, pp. 431–440, 2007.
- [18] G. Chen, O. van Gaans, and S. Verduyn Lunel, "Fixed points and pth moment exponential stability of stochastic delayed recurrent neural networks with impulses," *Applied Mathematics Letters. An International Journal of Rapid Publication*, vol. 27, pp. 36–42, 2014.
- [19] C. Guo, D. O'Regan, F. Deng, and R. P. Agarwal, "Fixed points and exponential stability for a stochastic neutral cellular neural network," *Applied Mathematics Letters. An International Journal of Rapid Publication*, vol. 26, no. 8, pp. 849–853, 2013.
- [20] X. Yang, Q. Zhu, and Z. Yao, "pth Moment Exponential Stability of Nonlinear Hybrid Stochastic Heat Equations," *Mathematical Problems in Engineering*, vol. 2014, Article ID 481246, 7 pages, 2014.
- [21] R. Rao and Z. Pu, "LMI-based stability criterion of impulsive T-S fuzzy dynamic equations via fixed point theory," *Abstract and Applied Analysis*, vol. 2013, Article ID 261353, 2013.
- [22] G.-Q. Wang and S. S. Cheng, "Fixed point theorems arising from seeking steady states of neural networks," *Applied Mathematical Modelling. Simulation and Computation for Engineering and Environmental Systems*, vol. 33, no. 1, pp. 499–506, 2009.
- [23] Y. Zhang and Q. Luo, "Global exponential stability of impulsive cellular neural networks with time-varying delays via fixed point theory," *Advances in Difference Equations*, vol. 2013, 23 pages, 2013.
- [24] R. Rao, S. Zhong, and Z. Pu, "LMI-based robust exponential stability criterion of impulsive integro-differential equations with uncertain parameters via contraction mapping theory," *Advances in Difference Equations*, vol. 2017, 19 pages, 2017.
- [25] J. Luo, "Exponentially stable stationary solutions for delay stochastic evolution equations," *Progress in Probability*, vol. 65, pp. 169–178, 2011.
- [26] A. A. Kwiecinska, "Stabilization of partial differential equations by noise," *Stochastic Processes and their Applications*, vol. 79, no. 2, pp. 179–184, 1999.
- [27] R. Rao and Z. Pu, "Stability analysis for impulsive stochastic fuzzy p-Laplace dynamic equations under Neumann or Dirichlet boundary condition," *Boundary Value Problems*, vol. 2013, 14 pages, 2013.
- [28] Q. Zhu, X. Li, and X. Yang, "Exponential stability for stochastic reaction-diffusion BAM neural networks with time-varying and distributed delays," *Applied Mathematics and Computation*, vol. 217, no. 13, pp. 6078–6091, 2011.
- [29] R. Rao, S. Zhong, and Z. Pu, "On the role of diffusion factors in stability analysis for p-Laplace dynamical equations involved to BAM Cohen-Grossberg neural network," *Neurocomputing*, vol. 223, pp. 54–62, 2017.
- [30] B. Xie, "The moment and almost surely exponential stability of stochastic heat equations," *Proceedings of the American Mathematical Society*, vol. 136, no. 10, pp. 3627–3634, 2008.
- [31] X. Li and F. Deng, "Razumikhin method for impulsive functional differential equations of neutral type," *Chaos, Solitons & Fractals*, vol. 101, pp. 41–49, 2017.

Research Article

Multiconstrained Network Intensive Vehicle Routing Adaptive Ant Colony Algorithm in the Context of Neural Network Analysis

Shaopei Chen,¹ Ji Yang,² Yong Li,² and Jingfeng Yang³

¹School of Public Administration, Guangdong University of Finance and Economics, Guangzhou, China

²Open Laboratory of Geo-Spatial Information Technology and Application of Guangdong Province, Guangzhou Institute of Geography, Guangzhou 510070, China

³Guangzhou Yuntu Information Technology Co., Ltd., Guangzhou 510532, China

Correspondence should be addressed to Yong Li; liyong@gdas.ac.cn and Jingfeng Yang; jingfengyang@126.com

Received 21 June 2017; Accepted 9 August 2017; Published 18 September 2017

Academic Editor: Yanan Li

Copyright © 2017 Shaopei Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Neural network models have recently made significant achievements in solving vehicle scheduling problems. Adaptive ant colony algorithm provides a new idea for neural networks to solve complex system problems of multiconstrained network intensive vehicle routing models. The pheromone in the path is changed by adjusting the volatile factors in the operation process adaptively. It effectively overcomes the tendency of the traditional ant colony algorithm to fall easily into the local optimal solution and slow convergence speed to search for the global optimal solution. The multiconstrained network intensive vehicle routing algorithm based on adaptive ant colony algorithm in this paper refers to the interaction between groups. Adaptive transfer and pheromone update strategies are introduced based on the traditional ant colony algorithm to optimize the selection, update, and coordination mechanisms of the algorithm further. Thus, the search task of the objective function for a feasible solution is completed by the search ants. Through the division and collaboration of different kinds of ants, pheromone adaptive strategy is combined with polymorphic ant colony algorithm. It can effectively overcome some disadvantages, such as premature stagnation, and has a theoretical significance to the study of large-scale multiconstrained vehicle routing problems in complex traffic network systems.

1. Introduction

Network intensive vehicle service, which is an important part of the complex urban public transport network system, is an innovation representative of the shared economy model in the era of Internet +. The Internet service platform has been attracting increasing attention in the shared economy era by creating a new commercial and management mode to adapt the Internet economy. It is also perceived as a representative of the development direction of the new economy and business model. Moreover, optimizing the network intensive vehicle scheduling model and improving the scheduling efficiency, response speed, and cost savings are important measures to improve the quality of network intensive vehicle services. Thus, studies on supply-and-demand information matching and of the path optimization algorithm for developing network intensive vehicle services

are of great significance. However, the vehicle scheduling problem on network intensive car rental is different from that of the classic vehicle scheduling problem. In the classic vehicle scheduling problem, the control center is assumed to know all the information related to optimal scheduling before the optimal scheduling instruction is executed, and the information does not change over time. In the era of Internet +, passenger demand information, such as location, time of demand, quantity demanded, travel time, and service time, is unknown and even dynamic before the path optimization. Furthermore, new information may have arrived or existing information may have changed. Therefore, dynamic vehicle routing problems (DVRPs) are more prominent in the network intensive vehicle scheduling problem. Moreover, the supply-and-demand information of network intensive vehicles is matched in real time seamlessly and dynamically and does not require manual docking and transmission. This

overcomes a problem that network booking taxi is forbidden to parade showmanship on the road under the existing management system. Therefore, the significant difference between the network intensive and classic vehicle scheduling problems is the continuous development of and change in the demand type and quantity in the former based on the real-time interaction through the Internet. In addition, real-time dynamic resource scheduling is performed according to the change in demand. However, the first principle of passenger transport in network intensive vehicle scheduling is to minimize the response time and relatively lessen the cost factors. This is also an important feature that distinguishes it from the traditional transportation problem and the vehicle routing problem (VRP). Network intensive vehicle scheduling problem is a multiconstrained path planning problem that integrates the dynamic network flow problem with multiconditioned and multi-start-and-stop points.

Studies on vehicle scheduling problem at home and abroad have gradually deepened in recent years. Studies on the target selection of modeling and consideration of constraints are mostly based on the simple extension of the traditional transportation problem and VRP. The single starting point and demand problem is extended to multiple starting points and demand problems. Moreover, the goal is extended from the single focus on cost minimization to the multiple objective optimizations of efficiency and cost [1–3]. Presently, no literature on the multiconstrained path problem of network intensive vehicles has been found, although some scholars have recently started to introduce some particular constraints into vehicle scheduling modeling. For instance, some scholars have established many kinds of dynamic vehicle scheduling models and designed concise and efficient heuristic algorithms for goods distribution or emergency material transportation vehicle scheduling problems [4–7]. Emergency material transportation vehicle scheduling models consider multitransport ways, multimaterials, multisupply points, and multidemand points, as they suppose that the number of vehicles, supply amount of materials and goods, and other parameters are dynamic. However, the goal of emergency material transportation is to minimize the total quantity of the goods that are not satisfied in the transport cycle. Although constraint is comprehensive in emergency material transportation vehicle scheduling models, vehicle scheduling under a dynamic demand still adopts the optimization method in a single cycle and limits the consideration of global optimization in multiple cycles [6, 7]. Other scholars similarly consider the vehicle dynamic scheduling problem under a vehicle shortage hypothesis and introduce minimum transportation time and cost into the goal. Nevertheless, the improvement of the vehicle scheduling efficiency is still limited to the optimal one in a single stage. The global consideration of dynamic vehicle scheduling problem is inadequate [8, 9]. Thus, many unsolved problems are elucidated in the research on network intensive vehicle scheduling problem. Therefore, considering the characteristics of DVRP with multiple constraints (DVRPMC), the construction and exploration of a simple intelligent algorithm with excellent

performance are important for the construction of a network intensive vehicle scheduling service system.

2. DVRPMC Description and Mathematical Model

2.1. Problem Description. A network intensive vehicle scheduling problem is oriented to the personal travel mode. Therefore, it should involve objective and subjective conditions and is a typical vehicle routing problem with multiple constraints, involving time, vehicle, traffic, safety, and so on. As a result, a network intensive vehicle schedule problem, that is, DVRPMC, should consider passenger priority in the order of booking time, demand of vehicle pattern, billing method, traffic conditions, time window, capacitance, safe surveillance, and other conditions. This is to ensure that network intensive scheduling is universal and practical. The detailed problem description is as follows. (1) A car is assigned to the passenger (i.e., customer point) according to the nearby principle. (2) The nearest dispatch point (i.e., parking spot of booking car) is selected from a total of K callable vehicles based on the time window limit to reach the customer point. If the vehicle arrives in advance, it must wait to serve the customer. (3) Different priorities are assigned according to passenger booking time. The highest grade is 1, and the lowest grade is R . High priority means that the passenger has significant weight. Thus, all aspects of priority should be satisfied. (4) Vehicles only consider passenger demands that are less than or equal to their carrying capacities. (5) Each passenger can select only one type of vehicle and charging method pattern and is only serviced by the selected car. (6) The passenger arrives at the target point and guides the vehicle to the nearest scheduled point. (7) For a transport service, the proper vehicle type, dispatch time, and path can minimize the time cost and obtain the highest response efficiency.

Information in DVRPMC changes over time. DVRPMC should consider time factors. Time axis is introduced here. The whole scheduling cycle (such as a working day) in a dynamic scheduling environment is constructed as the time axis. The time each new demand triggers a cycle is t . At t , demand information is divided into four kinds according to the state of the vehicle: (1) vehicles that have completed the task, (2) vehicles in service or on the way to serve, (3) vehicles that have responded to the service demand but do not proceed to the demand spots, and (4) vehicles waiting for new demands. A series of key points can be constructed on the time axis to represent the dynamic problem, and a vehicle scheduling system can arrange a scheduling plan according to the identified key points. The following scheduling strategies are adopted. (1) The vehicle from a certain scheduling point returns to the nearest scheduling point at the end of the delivery. (2) The fairness principle is followed to prevent customer points from obtaining any vehicle service response. Each customer service response shall not be lower than the level of its own demand. (3) The goal of the network intensive vehicle transport service is to minimize the response time, that is, from the initial start of the demand to the time when the vehicle reaches the demand point.

Compared with traditional VRP, the essence of DVRPMC is to create a vehicle scheduling plan for each customer point in the case of a known customer point distribution and the real-time development of any customer point. Moreover, many different vehicle models should be considered in DVRPMC. Thus, dynamic model selection and path arrangement are performed according to the passenger capacities of the models, the effect on the path, and the model requirements of passengers.

2.2. Mathematical Model. The mathematical model needs to satisfy the dynamic scheduling description. Therefore, the analysis mechanics can be described as follows: Firstly, this model is set to be $W(t) = \{w^{(t)}_i, i = 1, 2, 3, 4\}$, which refers to the set of all key points at t (Class 4). It is used to represent the dynamic problem by constructing a series of key points on the time axis. $w^{(t)}_1$ refers to the completed transport task, where the vehicle returned to the nearest scheduling point (parking point). $w^{(t)}_2$ refers to a passenger being served or a vehicle on the way to the service. $w^{(t)}_3$ refers to the vehicle that responded to the service demand but is not yet at the customer point. $w^{(t)}_4$ represents a vehicle in waiting state. Class 2 in the implementation of the task cannot be changed. The other three classes, as schedulable network intensive vehicles, are recorded as $K \in W(t)$ vehicles scheduled from the scheduling point (parking point) to n customers at t . Then, the scheduling task can be represented by a weighted graph $G^{(t)}(V, E)$ at t , where $V = (0, 1, 2, \dots, n)$ refers to a set of nodes, 0 refers to the scheduling point, n nodes refer to customer points, and $E = \{d_{ij}, i, j \in V\}$ refers to a set of paths from i to j . Last, considering road conditions and various factors, a coefficient of road condition λ_{ij} is used to represent the influence of road conditions on the scheduled vehicles. For the standard path, $\lambda_{ij} = 1$. If λ_{ij} is better than the standard path, $\lambda_{ij} < 1$; otherwise, $\lambda_{ij} > 1$. The coefficient of the road condition is multiplied by the actual length of the path between distribution points. d_{ij} is equal to the equivalent path length in consideration of the path influence. L refers to a feasible path. $f(L)$ refers to the cost corresponding to this path, that is, objective function. q refers to the quantity booked by the passenger. Q_k refers to the maximum load of vehicle K . The importance of the priority of a customer point is represented by weight σ_i . The time window of passenger i is $[a_i, b_i]$. The model demand of passenger i is G_i . The billing method of passenger i is F_i . t_{ij}^k refers to the travel time of vehicle K from the scheduling point to the customer point. The time vehicle K served at the starting point of passenger i is s_i . s_i^k is a decision variable that represents the moment vehicle K reaches customer point i . X_{ij}^k is also a decision variable and can be represented as follows:

$$X_{ij}^k = \begin{cases} 1 & \text{vehicle } k \text{ from } i \text{ to } j \\ 0 & \text{others.} \end{cases} \quad (1)$$

DVRPMC aims to determine a set of paths. The travel time of each path should be minimal, and no vehicle should exceed its carrying capacity Q_k . A feasible solution under a

different priority, model demand, different billing method, and road conditions is sought for highly efficient vehicle scheduling at minimum cost.

First, considering the cost of vehicle scheduling, C_0^k refers to the fixed cost of using vehicle K . C_1^k refers to the operating cost of vehicle K in the unit distance. V_k^{SP} refers to the travel speed of vehicle K under standard road conditions. Scheduling cost mainly includes the fixed cost of vehicle K ${}^v C_0^k$ and operating cost ${}^v C_1^k$. ${}^v C_0^k = C_0^k \sum X_{ij}^k$ ($i, j = 0, 1, 2, \dots, n; n \in k$), and ${}^v C_1^k = C_1^k \sum X_{ij}^k \lambda_{ij} d_{ij}$ ($i, j = 0, 1, 2, \dots, n; n \in k$). Second, the cost for waiting for customer s_i^k is considered, and $s_i^k = \mu s_i^k$. μ refers to the wait cost in the unit of time. Finally, extra cost C_i^P is incurred if vehicle K arrives beyond the time window of the passenger. To set C_i^P linear increase

$$C_i^P = c_i^1 \max[(a_i - s_i^k), 0] + c_i^2 \max[(s_i^k - b_i)], \quad (2)$$

where c_i^1 refers to the wait cost of vehicle K in customer point i with priority r per unit time and c_i^2 refers to the penalty per unit time if vehicle K arrives after the time window of the customer with priority r .

Therefore, the minimum objective function can be represented as

$$\min f(L) = \sum_{k=1}^k ({}^v C_0^k + {}^v C_1^k + s_i^k) + \sum_{i=1}^n C_i^P. \quad (3)$$

Constraint conditions are as follows:

$$\sum_{j=0}^n \sum_{i=0}^n X_{ij}^k q_j \leq Q_k \quad k = 1, 2, \dots, k \quad (4)$$

$$\sum_{i=0}^n \sum_{k=1}^k X_{ij}^k = 1 \quad j = 0, 1, 2, \dots, n \quad (5)$$

$$\sum_{j=1}^n X_{0j}^k = 1 \quad k = 1, 2, \dots, k \quad (6)$$

$$\sum_{i=1}^n X_{i0}^k - \sum_{j=1}^n X_{0j}^k = 0 \quad k = 1, 2, \dots, k \quad (7)$$

$$\sum_{i=1}^n X_{i0}^k = 1 \quad k = 1, 2, \dots, k \quad (8)$$

$$s_i + s_i^k + t_{ij}^k = s_j^k \quad (9)$$

$$\lambda_{ij} d_{ij} = V_k^{SP} t_{ij}^k \quad (10)$$

$$X_{ij}^k = \begin{cases} 1 & \text{vehicle } K \text{ from } i \text{ to } j \\ 0 & \text{others.} \end{cases} \quad (11)$$

Equation (3) refers to the objective function, which represents the minimum scheduling cost. Equation (4) refers to the sum of load capacities on a feasible path, which is not

more than the maximum load of the vehicle. Equation (5) refers to a vehicle (vehicle model and billing method that satisfies the requirements) that is distributed once in each customer point. Equations (6)–(8) refer to the subpath of each vehicle. A vehicle starts from the scheduling point, arrives at the customer point to provide transport service, brings passengers to the objective point, and finally returns to the scheduling point. Equation (9) indicates that if a vehicle travels directly from node i to j , the arrival time at node j is equal to the sum of the arrival time node i , service time, and travel time from node i to j . Equation (10) indicates that path distance is equal to the product of standard speed and travel time under the influence of road conditions. Equation (11) refers to the decision variable X_{ij}^k , which is $\{0, 1\}$.

3. Design of Optimization Algorithm of DVRPMC Based on Adaptive Ant Colony Algorithm

Intelligent algorithms have made great achievements in solving vehicle scheduling problems, especially for adaptive ant colony algorithm. Adaptive ant colony algorithm provides not only a new idea for solving the complex combination of optimization problems, but also a scientific perspective to deal with VRPs with multiple constraints. The pheromone in the path is changed by adaptively adjusting the volatile factors in the operation process. It effectively overcomes the tendency of the traditional ant colony algorithm to fall easily into the local optimal solution and slow convergence speed to search for the global optimal solution [10, 11]. Considering DVRPMC, this paper proposes an adaptive ant colony algorithm to introduce adaptive transfer and pheromone update strategies based on a traditional ant colony algorithm to optimize the selection, update, and coordination mechanisms of the algorithm further. Thus, considering the diversity of the ant colony, a task with many constraint conditions is provided to reconnaissance ants. Therefore, the search task for the feasible solution for the objective function is completed by the search ants. Through the division and collaboration of different kinds of ants, pheromone adaptive strategy is combined with polymorphic ant colony algorithm to effectively overcome some disadvantages, including long computing time and susceptibility to premature stagnation.

3.1. Principle and Implementation of Basic Ant Colony Algorithm. Ant colony algorithm is proposed based on the research on the real ant colony behavior in the natural world. It is a kind of simulated evolutionary algorithm based on population and belongs to the random search algorithm. Dorigo et al. [12, 13] first proposed this method and made full use of the similarities between the food search process of an ant colony and the famous traveling salesman problem

(TSP) to determine the shortest path from the ant nest to the food source through information exchange and cooperation among individuals to solve the TSP [14–18]. The principle and the method of the basic algorithm are as follows.

M ants are placed on N nodes selected at random. An ant selects the next node or cycle it has not yet visited based on a criterion, $\tau_{ij}(t)$, which is the concentration of residual information on the path from node i to node j at t . This is the information provided by the algorithm itself, and η_{ij} is the initial information from node i transferred to node j . This initial information is provided with the problem to be solved. $\eta_{ij} = 1/d_{ij}$ refers to an a priori value of node i to j . Thus, the probability that ant K at node i selects node j as the objective node at t is

$$P_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha(t) \eta_{ij}^\beta(t)}{\sum_{n \in \text{allowed}_k} \tau_{in}^\alpha(t) \eta_{in}^\beta(t)}, & j \in \text{allowed}_k \\ 0 & \text{others,} \end{cases} \quad (12)$$

supposing that $j \in N_i^k$.

α is the relative importance of the residual information and β is the relative importance of the expected value.

N_i^k refers to all possible objective nodes. Nodes mean no access. Each ant maintains a list tabu_k , which records all cities it has visited up to the present, to avoid multiple visits to the same node. P_{ij}^k refers to the probability that ant K transfers from node i to node j .

After each ant has accessed all n nodes (i.e., a cycle), residual information must be updated and old information must be weakened to prevent residual information from inundating inspired information caused by excessive residual information. New information on the ant access path must be added to τ_{ij} .

$$\tau_{ij}(t+n) = \rho \tau_{ij}(t) + \sum_{k=1}^K \tau_{ij}^k. \quad (13)$$

ρ refers to the reserve part of residual information. $1 - \rho$ refers to the weakened part of the residual information. ρ must be less than 1 to prevent the unlimited accumulation of information. $\Delta \tau_{ij}^k$ refers to the residual information content ant K left in the path from i to j in the access period t to $(t+n)$.

Gambardella and Dorigo et al. provided the Ant-Q (Quantity) algorithm model based on the basic ant colony algorithm [12]. Ant-Q algorithm is a reinforcement learning algorithm based on the Cellular Computing paradigm. Ant-Q's results on the vehicle routing problem, which is conceptually similar to fuel reload, are better than other ant colony optimization algorithms, such as Ant-C (Cycle) and Ant-D (Density) or the genetic algorithm (GA).

$$\Delta \tau_{ij} = \begin{cases} \frac{Q}{L_k}, & \text{the pheromone ant } K \text{ left in the period from } t \text{ to } t+1 \\ 0 & \text{otherwise.} \end{cases} \quad (14)$$

Otherwise, if ant K selects the path (i, j) in the period from t to $(t + n)$, Q is a constant, and L_k refers to the total path length ant K selected in this cycle. If it does not select this path,

$$\Delta\tau_{ij} = 0. \quad (15)$$

According to the Ant-Q algorithm concept, each step (i.e., from t to $(t + 1)$) requires the update of the concentration of residual information, but not the update of the residual information concentration after waiting for all ants to complete the access to all n cities. Therefore, in the Ant-Q algorithm model, $\Delta\tau_{ij} = Q/d_{ij}$, where d_{ij} refers to the distance from node i to node j and Q/d_{ij} is the concentration of residual information; that is, the concentration of residual information increases with the decrease of the distance to a city.

Therefore, the basic ant colony algorithm is mainly based on the combination of the principle of positive information feedback and a certain heuristic algorithm. This algorithm employs the random selection strategy in the process of constructing the solution. This selection strategy slows down the evolution. The positive feedback principle is designed to enhance the performance of the solution but is prone to stagnation. This is the root of the deficiency of the ant colony algorithm. Therefore, improving the selection strategy is necessary; that is, the selection strategy should be adapted based on the combination of deterministic and random selections. The evolution to a certain algebra dynamically adjusts the probability of deterministic selection in the search process. Thus, the evolution direction is basically determined. Dynamic adjustment at a particular time is based on the amount of information on the path [5]. Therefore, adaptive ant colony algorithm has increasingly attracted attention. This algorithm shrinks the gap in the quantity of information between the best and worst paths and appropriately enlarges the probability of random selection to conduct a complete search of the solution space with less than l to overcome the shortcomings of the basic ant colony algorithm effectively.

3.2. Principle and Design of Adaptive Ant Colony Algorithm.

Drawing lessons from the polymorphic ant colony method of Yang et al. [1], this paper divides ants in the ant colony algorithm into two types, namely, reconnaissance and search ants, because of the multiconstrained and dynamic characteristics of network intensive vehicle routing problems. Reconnaissance ants complete local reconnaissance and search ants complete global search.

Reconnaissance ants are endowed with multiconstrained tasks by the polymorphic ant colony method. Each customer point is regarded as the center to make local reconnaissance. Furthermore, a reconnaissance element is used to mark the reconnaissance result, in order to provide supplementary

information for a search ant, and select the next customer point after arriving at this point. The local search method of the reconnaissance ant colony places n reconnaissance ants in n passenger demand points, and each reconnaissance ant regards its location as the center from which to scout the feasibility of $n - 1$ points. The greater the feasibility, the higher the reconnaissance element on the path. Moreover, the reconnaissance element consists of three parts: (1) capacity-constrained reconnaissance element with a weight coefficient of ω_1 , (2) time window matching reconnaissance element with a weight coefficient of ω_2 , and (3) reconnaissance element in several points nearest to reconnaissance center (i.e., 20 points are taken from 100 demand points) with a weight coefficient of ω_3 , and $\omega_1 + \omega_2 + \omega_3 = 1$, where $0 < \omega_1, \omega_2, \omega_3 < 1$. If the capacity constraint is satisfactory, the contribution to the reconnaissance element is recorded as $\omega_1 C_1$. If it is unsatisfactory, the contribution to the reconnaissance element is 0. The matching factor of the time window in the routing starting point determines the reconnaissance element contributed by the time window matching degree $\omega_2 \varepsilon_{ji} C_2$. Matching factor $\varepsilon_{ji} = L([a_{ji}, b_{ji}], [a_i, b_i]) / (b_i - a_i)$ refers to the matching degree of the time window, and $a_{ji} = a_j + s_j + t_{ji}$. $L([a_{ji}, b_{ji}], [a_i, b_i])$ refers to the length of the overlapping parts of the two time windows. Under a fixed value of $[a_{ji}, b_{ji}]$, ε_{ji} is high if the overlapping part is large. That is, point j , which is the precursor of i , is reasonable from the angle of the time window. Passenger priority factors are considered in the search ant pheromone; that is, the factors are considered in the objective function but are not considered in the reconnaissance element. Moreover, to reduce the scout scope of reconnaissance ants, the statistic result of Quan and Wen [14] shows that the next node of a point is selected in several points nearest to this point in the path optimization solution without calculating all the remaining viable nodes. Thus, the optimization process can be accelerated and the solution is quite reliable, which can converge to the optimal solution with the probability of approaching 1. The reconnaissance result in ascending order is combined with the existing prior knowledge (integrated with max(PC) [14]) to generate another reconnaissance element, which is recorded as $\omega_3 \delta_{ij}$, on the path from point i to j . δ_{ij} ($i, j = 0, 1, 2, \dots, n-1; i \neq j$) is represented as follows:

$$\delta_{ij} = \begin{cases} \frac{\min d_{ix}}{d_{ij}} & \text{vehicle } k \text{ from } i \text{ to } j \\ 0 & \text{otherwise,} \end{cases} \quad (16)$$

where $\min d_{ix}$ refers to the nearest distance from city i as the center to other cities $n - 1$. To synthesize the total reconnaissance element of the above reconnaissance ants,

$$S_{ij} = \begin{cases} \xi_1 C_1 + \xi_2 \varepsilon_{ij} C_2 + \xi_3 \min \left(\frac{d_{ix}}{d_{ij}} \right) \varepsilon_{ji} \neq 0 & j \text{ and } i \text{ in max (PC)} \\ 0 & \text{others.} \end{cases} \quad (17)$$

Considering the need for a certain pheromone on the path at the initial moment, the amount of information of each path at the initial moment is set as

$$\tau_{ij}(0) = \begin{cases} CS_{ij} & S_{ij} \neq 0 \\ C(\min d_{ix}, \max d_{ix}) & S_{ij} = 0, \end{cases} \quad (18)$$

where $\min d_{ix}$ and $\max d_{ix}$ refer to the minimum and maximum distances from point i , as the center, to other points $n-1$. C refers to the concentration of pheromone on each path at the initial moment and is a constant. Through the reconnaissance element trail marked by the reconnaissance ants, research ants can make a directional search with the assistance of the pheromone. This can improve efficiency and determine the optimal solution.

The global search of the search ant colony draws lessons from the concept of the Ant-Q algorithm of Dorigo et al. [13] where the selection strategy is combined with the deterministic and random selection [1, 4]: (1) K is the number of search ants, η_{ij} refers to the visibility of side (i, j) , and $\eta_{ij} = 1/f(d_{ij})$, which reflects the heuristic degree transfer from node i to j and is different from $\eta_{ij} = 1/d_{ij}$ in the basic ant

$$P_{ij}^k = \begin{cases} Q_1 \frac{\tau_{ij}^\alpha \eta_{ij}^\beta}{\sum_{x \in a_k} \tau_{ij}^\alpha \eta_{ij}^\beta} + Q_2 \frac{\theta_j / (|s_j^k - a_j^k| + |s_j^k - b_j^k|)}{\sum_{x \in a_k} \theta_x / (|s_x^k - a_x^k| + |s_x^k - b_x^k|)} & j \in a_k \\ 0 & \text{others,} \end{cases} \quad (19)$$

where a_k refers to the set of the service points and destinations that ant K is allowed to select for the next step; α and β parameters reflect the relative importance the pheromone ants accumulated during the movement and heuristic information provided that the ants select the path, respectively; Q_1 and Q_2 are weight coefficients that satisfy $0 \leq Q_1 \leq 1$, $0 \leq Q_2 \leq 1$, $Q_1 + Q_2 = 1$, and $0 \leq \theta_j \leq 1$, which refers to the taboo list (tabu_k) each ant creates, where $k = 1, 2, \dots, K$ (K refers to the sum of search ant colony) for recording the nodes that ant K has visited at t and the ants are forbidden to visit again in this cycle. The tabu list is cleared at the end of the cycle.

3.3. Adaptive Information Update Strategy. Traveling ants often cause blockage and stagnation. Ant colony algorithm easily leads to premature and local convergences. To solve this problem and improve the global convergence of ant colony algorithm and search speed, many studies proposed different strategies to update existing information [19–25]; for example, in the standard ant colony algorithm, when information is updated and as long as the ant is traversed, selecting the path can update the information on the path. This enhances the information on the path for optimal fitness and weakens the information on other paths; the other algorithms based on level change enable the ant to set several paths with relatively good fitness, and the degree of excellence of its solution determines the magnitude of information.

colony algorithm model. The objective function is placed in the global scope given the best visibility. It reflects the big-picture thinking of ants and does not necessarily have to select the closest point in a large probability as the next point. However, each step of the ant must consider the objective function of overall optimization. (2) τ_{ij} refers to the pheromone trail intensity of side (i, j) , P_{ij}^k refers to state mobility probability of ant K from node i to node j , and j refers to the node that is not yet visited. When each ant selects the next node, two points should be considered while selecting the next service point under the premise of adhering to the vehicle capacity and time window constraints. One is visibility and the amount of information access to the next service point. Second is priority conditions priority-of-small time window and priority-of-shortest waiting time.

The search ant colony is tasked to conduct a global search. At each customer point, reconnaissance element and the pheromone at each side select the next service point until they find and mark the best route. The formula for the probability P_{ij}^k of the search ant colony k ($k = 1, 2, \dots, n$) that transfers from i to j at t during the movement is as follows:

These algorithms presented above are different. They mainly update the amount of information using the increasing or decreasing proportion of the fixed amount of information and ignore the distribution features of the solution. They improve the characteristics of the ant colony algorithm to a certain extent and are used to treat small-scale problems. This paper proposes a new adaptive information update strategy for solving the large-scale problems, which starts from the distribution state of the solution.

In large-scale problems, the existence of a volatile coefficient reduces the amount of information on the path that has never been searched to be close to 0, which lowers the search ability of the algorithm on these paths. If the amount of information on a path is large, the amount of information on these paths is increased, and the chance that the path that has been searched is selected again is high. Moreover, the global search ability of the algorithm is influenced. A fixed changing volatile coefficient can improve the global search ability but reduces the convergence speed of the algorithm. Therefore, an adaptive method for changing τ is proposed to update the pheromone. The formula is as follows [11]:

$$\tau_{ij}(t+1) = (1 - \rho) \tau_{ij} + \Delta \tau_{ij}; \quad (20)$$

while $\tau < \tau_{\min}$,

$$\tau_{ij}(t+1) = (1 - \rho)^{1+\phi(m)} \tau_{ij}(t) + \Delta \tau_{ij}; \quad (21)$$

while $\tau > \tau_{\max}$,

$$\tau_{ij}(t+1) = (1 - \rho)^{1-\phi(m)} \tau_{ij}(t) + \Delta\tau_{ij}, \quad (22)$$

where $\phi(m)$ is a function proportional to the number of convergences m . If the number of convergences m is higher, $\phi(m)$ is larger, as denoted by the equation below:

$$\phi(m) = \frac{\text{Continuous convergence times } m}{c}, \quad (23)$$

where c is a constant. The distribution of the solution adaptively updates the amount of information to dynamically adjust the intensity of information on each path, causing the moderate concentration or scattering of ants to avoid premature convergence and local convergence and improving global search ability. The adaptive information update strategy adaptively updates the amount of information based on the distribution of the solution. Thus, it dynamically adjusts the intensity of information on each path, increases the diversity of the solution space, improves the global search ability, and prevents premature convergence and local convergence.

4. Case Analysis

The DVRPMC optimal model and algorithm based on ant colony algorithm have many constrained conditions and parameters. To verify the performance of the algorithm, the customer point priority, service time of customer point, coefficient of road condition, vehicle wait cost, and other constrained conditions have not yet been considered in this case analysis. The objective function in the model is also simplified as the path distance. In this case analysis, 15 customer points were distributed in a square region in the edge length of 8 km. Table 1 illustrates the customer points' coordinates and related demand for passenger capacity, respectively. The region has two scheduling points (parking points). The coordinates are $(-2, -2)$ and $(3, 3)$, which own five and eight vehicles, respectively. The maximum passenger capacity of each vehicle is four.

The VRP standard case library of Solomon and Desrosiers [26] is used for detection. The Solomon and Desrosiers case library is divided into three categories: Class C, Class R, and Class RC. The customer points in Class C are distributed in cluster types. The customer points in Class R are distributed at random. The customer points in Class RC are the mixed-half cluster type. This case analysis is conducted in the hardware environment of Intel Pentium IV 3.0 with 1 G memory. VC language is used for the programming of the algorithm and to calculate R and C. The result is compared with the optimal solution from several existing algorithms. The parameters of the algorithm are $\alpha = 1$, $\beta = 2$, and $\rho = 0.8$. The iteration number is 50. The results of 10 times the calculation are shown in Table 2.

Table 2 shows that the optimal solutions of RC and R are mostly in the twenties. The calculation time is not more than 16 s in the hardware environment of Intel Pentium IV 3.0 with 1 G memory. Thus, the optimal solution convergence is faster than that of the standard ant colony algorithm. To

TABLE 1: Coordinates of customer points and demand for passenger capacity (DPC).

Number	Coordinates	DPC
(1)	0, -1.3	2
(2)	0, 1.5	1
(3)	-2.1, -2.1	3
(4)	3.9, 3.1	4
(5)	-2.8, -1.1	2
(6)	-2.3, 0.2	1
(7)	2.1, 0	1
(8)	1.9, -1	3
(9)	0.96, -3.2	3
(10)	-2.5, -1.7	2
(11)	-2.98, 1.4	1
(12)	1.97, 0.9	4
(13)	2.0, 1.9	2
(14)	-2.9, -1.7	2
(15)	-3.8, 2.4	2

compare the calculation results, standard genetic, adaptive genetic, and ant colony optimal adaptive algorithms are used to stimulate the case study. The obtained parameters in the case of 10 search times for the Class RC are shown in Table 3.

To compare the calculation results, the average number of vehicles in Table 3 up to two digits after the decimal point is taken. The calculation result shows that, in Class RC, the DVRPMC calculation results of the average number of vehicles and traveling distances are the best. Thus, the calculation result of the adaptive multiconstrained ant colony algorithm is stable with short traveling distance and high calculation efficiency.

5. Conclusion

In the context of neural network analysis, this paper analyzes the existing ant colony algorithm and proposes a multiconstrained network intensive vehicle routing optimal model and algorithm based on the adaptive ant colony algorithm. Drawing lessons from the adaptive polymorphic ant colony algorithm, the adaptive multiconstrained network intensive vehicle ant colony algorithm in this paper is used to collaboratively solve the problems through different types of ants. It overcomes the shortcomings of the traditional ant colony algorithm of having one ant colony with low algorithm efficiency. The ant colony can exchange planning information during the problem-solving process and maintain diversity in the search process because of the interaction among populations. Therefore, through the further use of the adaptive information update strategy, the algorithm of introducing the interaction among populations performs better than the traditional algorithm. Finally, this paper conducts a simulation experiment using specific examples. The experimental result shows that the adaptive multiconstrained ant colony algorithm has better solving efficiency and results than the standard ant colony algorithm. Therefore, for large-scale DVRPMC in network intensive vehicle

TABLE 2: Experimental results.

Number	RC		R	
	Optimal solution	Iteration number	Optimal solution	Iteration number
(1)	1526	18	1488	20
(2)	1577	25	1501	19
(3)	1456	21	1401	32
(4)	1411	19	1389	29
(5)	1511	24	1435	33
(6)	1523	30	1367	35
(7)	1489	33	1478	28
(8)	1456	31	1501	21
(9)	1501	29	1459	34
(10)	1531	20	1347	27

TABLE 3: Comparison of the calculation results of various algorithms.

Parameters	RC		
	SGA	ASGA	DVRPMC
Number of vehicles	4.98	4.25	3.32
Iteration number	43.2	45.6	27.8
Traveling distance	1582.47	1475.60	1409.70

scheduling, the adaptive multiconstrained and colony algorithm has significant superiority. Through the division and collaboration of different kinds of ants, pheromone adaptive strategy is combined with polymorphic ant colony algorithm; the algorithm given in this paper can effectively overcome some disadvantages, such as premature stagnation, and has a theoretical significance to the study of large-scale multi-constrained vehicle routing problems in complex traffic network systems. The established general mathematical model can be used in future studies, involving verifying further its result and efficiency in the case study of simulation under different passenger priorities, different road conditions, spatiotemporal traffic networks, and other cases' study of the optimal routing search method of real-time network intensive vehicle scheduling in a dynamic environment intensively. Furthermore, the algorithm provided in this paper should be significant for the further study on routing problem in the dynamic environment of a spatial and temporal complexity traffic network.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This study was supported by the Guangdong Natural Science Fund (2015A030313626), Special Fund of Frontier and Key Technology Innovation of Guangdong Province (Provincial Major Science and Technology Project) (2014B010112008,

2015B010106001, 2015B010129003, and 2016B010109007), Pearl River S&T Nova Program of Guangzhou (201610010034), Implementation of Innovation Driven Development Capacity Building Special Funds of Guangdong Academy of Sciences (2017GDASCX-0101, 2017GDASCX-0601), Guangdong Science and Technology Program (2013B01010201, 2013B010102018, and 2013B090600152), and Guangzhou Science and Technology Project (2014Y2-00044).

References

- [1] C. Yang, Z. Li, and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Transactions on Cybernetics*, vol. 43, no. 1, pp. 24–36, 2013.
- [2] A. Garcia-Najera and J. A. Bullinaria, "An improved multi-objective evolutionary algorithm for the vehicle routing problem with time windows," *Computers & Operations Research*, vol. 38, no. 1, pp. 287–300, 2010.
- [3] S. F. Ghannadpour, S. Noori, R. Tavakkoli-Moghaddam, and K. Ghoseiri, "A multi-objective dynamic vehicle routing problem with fuzzy time windows: model, solution and application," *Applied Soft Computing Journal*, vol. 14, no. 1, pp. 504–527, 2014.
- [4] M. Chen, Z. Zhang, and J. Shi, "Vehicle routing problem with multiple constraints using adaptive and polymorphic ant colony algorithm," *Journal of Southeast University (Natural Science Edition)*, vol. 38, no. 1, pp. 37–42, 2008.
- [5] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural Network-based Motion Control of An Under-actuated Wheeled Inverted Pendulum Model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, pp. 2004–2016, 2014.
- [6] H. Wang, B. Li, and K. Liu, "Demand allocation and network flow assignment under emergency rescue circumstance," *Systems Engineering-Theory & Practice*, vol. 35, no. 6, pp. 1457–1464, 2015.
- [7] H. Xiao, Z. Li, C. Yang et al., "Robust stabilization of a wheeled mobile robot using model predictive control based on neurodynamics optimization," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 1, pp. 505–516, 2017.
- [8] K. Li, L. Tang, and S. Chen, "Modeling and optimizing for the integrated problem with container storage allocation and truck scheduling," *System Engineering-Theory & Practice*, vol. 34, no. 1, pp. 115–121, 2014.

- [9] Y. Chen and J. Wang, "Improved Ant Colony Algorithm for Vehicle Routing Problem with Time Windows," *Journal of Computer Engineering and Applications*, vol. 43, no. 29, pp. 218–219, 2006.
- [10] L. Shao, "Ant Colony Optimizing Adaptive Gene Algorithm Resolving Logistics Vehicle Schedule," *Computer Measurement & Control*, vol. 20, no. 5, pp. 1423–1425, 2012.
- [11] Y. Wang and J. Xie, "An adaptive ant colony optimization algorithm and simulation," *Journal of System Simulation*, vol. 14, no. 1, pp. 31–33, 2014.
- [12] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [13] M. Dorigo and M. Luca, "A study of some properties of Ant-O," Tech. Rep. TR/IRIDIA/1996-4, IRIDIA, Universit Libre de Bruxelles, 1996.
- [14] H. Quan and G. Wen, "Subspacegenetical algorithm for TSP," *Mathematical Theory and Applications*, vol. 22, no. 1, pp. 36–39, 2002.
- [15] A. G. Qureshi, E. Taniguchi, and T. Yamada, "An exact solution approach for vehicle routing and scheduling problems with soft time windows," *Transportation Research Part E: Logistics & Transportation Review*, vol. 45, no. 6, pp. 960–977, 2009.
- [16] I. H. Osman, "Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem," *Annals of Operations Research*, vol. 41, no. 4, pp. 421–451, 1993.
- [17] J. Schulze and T. Fahle, "A parallel algorithm for the vehicle routing problem with time window constraints," *Annals of Operations Research*, vol. 86, pp. 585–607, 1999.
- [18] A. S. Kenyon and D. P. Morton, "Stochastic vehicle routing with random travel times," *Transportation Science*, vol. 37, no. 1, pp. 69–82, 2003.
- [19] Q. Cao, X. Liu, and X. Ren, "Vehicle schedule problem based on plant growth simulation algorithm," *System Engineering-Theory & Practice*, vol. 35, no. 6, pp. 1449–1456, 2015.
- [20] M. Wei and W. Jin, "Discrete particle swarm optimization algorithm for vehicle scheduling problem," *Computer Science*, vol. 37, no. 4, pp. 187–191, 2010.
- [21] Z. Zhao, Y. Liu, and F. Luo, "Output feedback boundary control of an axially moving system with input saturation constraint," *ISA Transactions*, vol. 68, pp. 22–32, 2017.
- [22] Z. Zhao, Y. Liu, F. Guo, and Y. Fu, "Vibration control and boundary tension constraint of an axially moving string system," *Nonlinear Dynamics*, vol. 89, no. 4, pp. 2431–2440, 2017.
- [23] Z. Zhao, Y. Liu, and F. Luo, "Boundary Control for a Vibrating String System with Bounded Input," *Asian Journal of Control*, In press.
- [24] Z. Zhao, Y. Liu, W. He, and F. Luo, "Adaptive boundary control of an axially moving belt system with high acceleration/deceleration," *IET Control Theory & Applications*, vol. 10, no. 11, pp. 1299–1306, 2016.
- [25] Z.-J. Zhao, Y. Liu, F. Guo, and Y. Fu, "Modelling and control for a class of axially moving nonuniform system," *International Journal of Systems Science. Principles and Applications of Systems and Integration*, vol. 48, no. 4, pp. 849–861, 2017.
- [26] M. M. Solomon and J. Desrosiers, "Time window constrained routing and scheduling problems," *Operations Research Society of America. Transportation Science Section. Transportation Science*, vol. 22, no. 1, pp. 1–11, 1988.

Research Article

Neural Network-Based State Estimation for a Closed-Loop Control Strategy Applied to a Fed-Batch Bioreactor

Santiago Rómoli,¹ Mario Serrano,¹ Francisco Rossomando,² Jorge Vega,^{3,4} Oscar Ortiz,¹ and Gustavo Scaglia¹

¹*Instituto de Ingeniería Química, Universidad Nacional de San Juan, CONICET, Av. Lib. San Martín Oeste, 1109 San Juan, Argentina*

²*Instituto de Automática, Universidad Nacional de San Juan, CONICET, Av. Lib. San Martín Oeste, 1109 San Juan, Argentina*

³*Facultad Regional Santa Fe, Universidad Tecnológica Nacional, CONICET, Lavaisse 610, Santa Fe, Argentina*

⁴*Instituto de Desarrollo Tecnológico para la Industria Química (INTEC (UNL-CONICET)), Güemes, 3450 Santa Fe, Argentina*

Correspondence should be addressed to Mario Serrano; eserrano@fi.unsj.edu.ar

Received 15 June 2017; Accepted 9 July 2017

Academic Editor: Chenguang Yang

Copyright © 2017 Santiago Rómoli et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The lack of online information on some bioprocess variables and the presence of model and parametric uncertainties pose significant challenges to the design of efficient closed-loop control strategies. To address this issue, this work proposes an online state estimator based on a Radial Basis Function (RBF) neural network that operates in closed loop together with a control law derived on a linear algebra-based design strategy. The proposed methodology is applied to a class of nonlinear systems with three types of uncertainties: (i) time-varying parameters, (ii) uncertain nonlinearities, and (iii) unmodeled dynamics. To reduce the effect of uncertainties on the bioreactor, some integrators of the tracking error are introduced, which in turn allow the derivation of the proper control actions. This new control scheme guarantees that all signals are uniformly and ultimately bounded, and the tracking error converges to small values. The effectiveness of the proposed approach is illustrated on the basis of simulated experiments on a fed-batch bioreactor, and its performance is compared with two controllers available in the literature.

1. Introduction

In the last decade, several control strategies applied to different bioprocesses have been extensively investigated (e.g., [1–5]). In the particular case of fed-batch bioreactors, some nonlinear control strategies were developed to improve the process efficiency and the “batch-to-batch” reproducibility [6], as well as to deal with the unavoidable uncertainties typically present in bioprocesses [7].

Many articles have investigated the effects of model uncertainties on the performance of model-based nonlinear control strategies typically applied to bioprocesses. For example, the “yield coefficient of cell to glucose” in a bioprocess is a time-varying parameter often assumed as a constant. Also, the “specific growth rate” can be represented through the Monod-Haldane model [8], but it is possible that the model becomes inaccurate because of yeast growth inhibition due to an eventual presence of ethanol. In principle,

three main types of uncertainties can be envisaged [9–13]: (1) uncertain and/or time-varying model parameters, (2) uncertain nonlinearities, and (3) unmodeled dynamics. Additionally, nonmodeled external perturbations can disturb the process. In general, model uncertainties and external perturbations can cause severe risks in bioreactors, such as leading to undesirable operating conditions [14–16]. For instance, inadequate modeling of the inhibitory effect of a substrate can lead to a structural instability in the dynamical behavior of a bioprocess [15].

Adaptive control strategies are a valuable choice to control bioreactors due to their ability to compensate for parametric uncertainties. The Nussbaum gain method has been applied to control a bioprocess of unknown and time-varying control gain and parameters [17]. A fed-batch bioreactor control with simultaneous dynamic identification of unknown state and parameters has been investigated through an algorithm based on the auxiliary model principle with adaptive controllers

[18]. A control strategy was proposed to track a desired profile for the biomass of a yeast culture in a high-density fed-batch bioreactor [19]. This last strategy was based on a model that included an uncertainty term estimated through an adaptive technique. Also, an adaptive nonlinear feedback controller has been designed to automatically estimate uncertain time-varying parameters in order to guarantee the closed-loop control of a waste water plant [20].

Fuzzy control systems have been successfully applied to several bioreactors. For instance, [12] considered an adaptive fuzzy-decentralized control for a system with unknown nonlinear uncertainties and dynamical uncertainties. The fuzzy system was used to estimate the nonlinear uncertainties. Also, a fuzzy controller for the alcoholic fermentation with *Saccharomyces cerevisiae* has been proposed [21]. It combined the capability of handling uncertainties with the ability of predictive control to improve the plant performance, making use of a neural network model of the nonlinear system. Optimal control strategies have also been applied to bioprocesses [22, 23]. In order to compensate for process uncertainties, this type of controllers incorporates an additional term to the bioprocess model (typically, in the specific growth rate), which is estimated through an optimization technique. Alternative strategies for controlling bioreactors under model uncertainties are trajectory-based control [24], hybrid control [25], and model predictive control [26]. Additionally, due to the inherent nature of a bioprocess, a control scheme based on a stochastic model seems to be adequate. The specific growth rate can be modeled as affected by a white noise representing environment uncertainties, and the mathematical model consists of a set of stochastic differential equations [27] (some applications can be consulted in [28, 29]). The application of this approach onto a real plant is quite difficult due to its complex design and implementation.

Based on a linear algebra methodology, a control strategy has been applied for trajectory tracking in a bioprocess [30]. The technique was used to calculate the control actions that ensure the tracking of independently determined optimal trajectories of some process variables. To this effect, a discrete approximation of the original process model must first be derived, and the control actions are then calculated through a least square problem. This methodology has been applied to a fed-batch bioprocess designed to produce a secreted protein, and it was originally modeled by Park and Ramirez [31]. Optimal profiles of such bioprocess have also been published [32].

Many nonlinear control techniques of bioprocesses assume that the feedback variables can be measured online [33, 34]. However, such hypothesis is often unrealistic and therefore some process variables must be estimated through virtual sensors (or soft sensors). There are several kinds of virtual sensors published in the literature which exhibit different abilities to deal with model mismatches and perturbations. In this work, an artificial intelligence-based observer has been chosen because this type of estimator is especially recommended for highly nonlinear systems in the presence of uncertainties [35].

This paper aims at designing a controller for the multivariable tracking of optimal profiles when the process is

affected by uncertainties in the dynamic model and/or in the model parameters. Unmeasured process variables are estimated through a RBF neural network. The proposed approach is evaluated onto a bioprocess originally modeled by [31], with the optimal profiles determined by [32] in absence of model errors. Brief descriptions of the fed-batch bioreactor and the controller derived in [30] are presented in Appendix A.

In comparison with [30], the proposed approach prevents the polynomial uncertainties from affecting the tracking control. To achieve this aim, an integrative term is incorporated in the linear algebra methodology originally proposed. Then, the conditions in order for a system of linear equations to have exact solution are analyzed. The control action is obtained by solving the linear system, even though the original process model is nonlinear.

The main contribution is that the original method proposed herein ensures the convergence to zero of the tracking errors under additive uncertainties. Another important contribution is the inclusion of RBF neural network estimators: the control issue is addressed considering more realistic conditions with nonmeasurable variables. Further, the stability analysis is made using Lyapunov theory in discrete time.

The article is organized as follows. In Section 2, the controller design methodology and the convergence to zero of the tracking errors are presented. Section 3 shows briefly the description of the RBF estimator. A method to tune the controller is presented in Section 4, along with four testing cases designed to check the effectiveness of the new control law. Main conclusions of the work are summarized in Section 5.

2. Controller Design

Derivation of the control strategy is based on the mathematical model presented in Appendix A. The effect of an additive uncertainty (\mathbf{E}_n) is modeled by extending (A.4) as follows:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{c}_n \Delta t - \mathbf{d}_n u_n \Delta t + \mathbf{E}_n, \quad (1)$$

where \mathbf{z}_n is the state vector, \mathbf{c}_n and \mathbf{d}_n are the input vectors, Δt is the sample time, u_n is the control action, and \mathbf{E}_n is an additive uncertainty. In principle, \mathbf{E}_n can be used to model either perturbed systems or a wide class of model mismatches. In what follows, a control law is designed to compensate for the tracking errors introduced by \mathbf{E}_n .

Taking into account the fact that the mismatch might depend on both system states and inputs, we here consider a real plant described by $\mathbf{z}_{n+1} = f(\mathbf{z}_n, \mathbf{u}_n)$. The additive uncertainty can be expressed as $\mathbf{E}_n = f(\mathbf{z}_n, \mathbf{u}_n) - \hat{f}(\mathbf{z}_n, \mathbf{u}_n)$, where $\hat{f}(\mathbf{z}_n, \mathbf{u}_n)$ is the discrete-time nonlinear system model. If \mathbf{z} and \mathbf{u} are assumed to be bounded and f is Lipschitz, then \mathbf{E}_n can be modeled as a bounded uncertainty [36–38].

With the additive uncertainty of (1), the fed-batch bioreactor model of (A.3) becomes

$$\begin{bmatrix} \frac{P_n}{V_n} \\ \frac{T_n}{V_n} \\ \frac{X_n}{V_n} \\ \frac{S_F - S_n}{V_n} \end{bmatrix} u_n^* = \begin{bmatrix} \chi_n (T_n - P_n) - \frac{P_{n+1} - P_n}{\Delta t} \\ \psi_n X_n - \frac{T_{n+1} - T_n}{\Delta t} \\ \mu_n X_n - \frac{X_{n+1} - X_n}{\Delta t} \\ 7.3 \mu_n X_n + \frac{S_{n+1} - S_n}{\Delta t} \end{bmatrix} \quad (2)$$

$$+ \begin{bmatrix} E_{P,n} \\ E_{T,n} \\ E_{X,n} \\ E_{S,n} \end{bmatrix}.$$

The control action for the secreted protein concentration, P , given by (A.4) of [30], is

$$u_{n,P} = \frac{\chi (T_n - P_n) - (P_{\text{ref},n+1} - k_p (P_{\text{ref},n} - P_n) - P_n) / \Delta t}{P_n / V_n}. \quad (3)$$

The discrete value of P , P_{n+1} , is calculated by replacing (3) into the first row of (2):

$$P_{n+1} = P_{\text{ref},n+1} - k_p (P_{\text{ref},n} - P_n) + \Delta t (T_n - P_n) (\chi_n - \chi) + \Delta t E_{P,n}, \quad (4)$$

where χ_n and χ depend on S_n and $S_{ez,n}$, respectively. Thus, the tracking error of P_n is

$$e_{P,n+1} = P_{\text{ref},n+1} - P_{n+1}. \quad (5)$$

By combining (4) and (5), one obtains

$$e_{P,n+1} = k_p e_{P,n} - \Delta t (T_n - P_n) (\chi_n - \chi) - \Delta t E_{P,n}. \quad (6)$$

The Taylor interpolation of χ_n at its desired value χ is

$$\chi_n = \chi + \frac{d\chi}{dS} \Big|_{S=S_{ez,n} + \lambda(S_n - S_{ez,n})=S_\lambda} (S_n - S_{ez,n}); \quad (7)$$

$$0 < \lambda < 1.$$

By replacing (7) into (6), the tracking error for P_n becomes

$$e_{P,n+1} = k_p e_{P,n} - \Delta t (T_n - P_n) \left[\chi + \frac{d\chi}{dS} \Big|_{S=S_\lambda} (S_n - S_{ez,n}) - \chi \right] - \Delta t E_{P,n}, \quad (8)$$

$$e_{P,n+1} = k_p e_{P,n} + \Delta t (T_n - P_n) \frac{d\chi}{dS} \Big|_{S=S_\lambda} (S_{ez,n} - S_n) - \Delta t E_{P,n}.$$

If a similar procedure is followed for T , X , and S , then the error equation becomes [30]

$$\mathbf{e}_{n+1} = \begin{bmatrix} e_{P,n+1} \\ e_{T,n+1} \\ e_{X,n+1} \\ e_{S,n+1} \end{bmatrix} = \begin{bmatrix} k_p & 0 & 0 & 0 \\ 0 & k_T & 0 & 0 \\ 0 & 0 & k_X & 0 \\ 0 & 0 & 0 & k_S \end{bmatrix} \begin{bmatrix} e_{P,n} \\ e_{T,n} \\ e_{X,n} \\ e_{S,n} \end{bmatrix} \quad (9a)$$

$$+ \Delta t \begin{bmatrix} (T_n - P_n) \frac{d\chi}{dS} \Big|_{S=S_\lambda} \\ X_n \frac{d\psi}{dS} \Big|_{S=S_\varphi} \\ X_n \frac{d\mu}{dS} \Big|_{S=S_\delta} \\ 0 \end{bmatrix} e_{S,n} - \begin{bmatrix} E_{P,n} \\ E_{T,n} \\ E_{X,n} \\ E_{S,n} \end{bmatrix}.$$

Alternatively, in vector form,

$$\mathbf{e}_{n+1} = \mathbf{K} \mathbf{e}_n + \mathbf{h}_n - \mathbf{E}_n, \quad (9b)$$

where \mathbf{h}_n is a nonlinearity that tends to zero when $e_{S,n}$ tends to zero. Equations (9a) and (9b) suggest a direct effect of the additive uncertainty on the tracking error. Therefore, the presence of an uncertainty avoids a convergence of tracking errors to zero (see Theorem 1 of [30]).

2.1. Single Integral Action. In order to reduce the effect of \mathbf{E}_n , some integrators for the tracking errors will be introduced, depending on the time variation hypothesis of \mathbf{E}_n . We will assume that \mathbf{E}_n is unknown and each of its components is an m -order polynomial.

Remark 1. The first-order difference of \mathbf{E}_n is defined as $\delta \mathbf{E}_n = \mathbf{E}_{n+1} - \mathbf{E}_n$, the second-order difference is defined as $\delta^2 \mathbf{E}_n = \delta(\delta \mathbf{E}_n) = \delta(\mathbf{E}_{n+1} - \mathbf{E}_n) = \mathbf{E}_{n+2} - 2\mathbf{E}_{n+1} + \mathbf{E}_n$, and, as a rule, the q -th order difference is defined as $\delta^q \mathbf{E}_n = \delta(\delta^{q-1} \mathbf{E}_n)$.

Remark 2. The q -th difference of a $q - 1$ -order polynomial is zero.

Consider first a constant uncertainty $\mathbf{E}_n = \text{const}$. Then, $\delta \mathbf{E}_n = \mathbf{E}_{n+1} - \mathbf{E}_n = \mathbf{0}$. In this case, an integrator for each state variable will force the error to converge to zero. Call $\mathbf{e}(t)$ the continuous time error in the state vector. Then, the integral of the tracking error is defined as

$$\mathbf{U}_{1,n+1} = \mathbf{U}_{1,n} + \int_{n\Delta t}^{(n+1)\Delta t} \mathbf{e}(t) dt \cong \mathbf{U}_{1,n} + \mathbf{e}_n \Delta t. \quad (10)$$

Hence, the control action (A.7) will be computed by assuming a new term in (A.5); that is,

$$\begin{aligned} \mathbf{z}_{n+1} &= \mathbf{z}_{\text{ref},n+1} - \mathbf{K}(\mathbf{z}_{\text{ref},n} - \mathbf{z}_n) + \mathbf{k}_1 \mathbf{U}_{1,n+1}, \\ \mathbf{k}_1 &= \begin{bmatrix} k_{P1} & 0 & 0 & 0 \\ 0 & k_{T1} & 0 & 0 \\ 0 & 0 & k_{X1} & 0 \\ 0 & 0 & 0 & k_{S1} \end{bmatrix}, \end{aligned} \quad (11)$$

where \mathbf{K} and \mathbf{k}_1 are the proportional and integral design parameters, respectively.

By adding one integrator ($\mathbf{k}_1 \mathbf{U}_{1,n+1}$) to (9b) and using (10), one obtains

$$\mathbf{e}_{n+1} + \mathbf{k}_1 (\mathbf{U}_{1,n+1} + \mathbf{e}_n \Delta t) = \mathbf{K} \mathbf{e}_n + \mathbf{h}_n - \mathbf{E}_n. \quad (12)$$

Rewriting (12) for the next sample time,

$$\mathbf{e}_{n+2} + \mathbf{k}_1 (\mathbf{U}_{1,n+2} + \mathbf{e}_{n+1} \Delta t) = \mathbf{K} \mathbf{e}_{n+1} + \mathbf{h}_{n+1} - \mathbf{E}_{n+1}. \quad (13)$$

Combining (10) and (12) and substituting them into (13),

$$\begin{aligned} \mathbf{e}_{n+2} + \mathbf{k}_1 \mathbf{e}_{n+1} \Delta t + \mathbf{h}_n - \mathbf{E}_n - \mathbf{e}_{n+1} + \mathbf{k}_1 \mathbf{e}_n \\ = \mathbf{K} \mathbf{e}_{n+1} + \mathbf{h}_{n+1} - \mathbf{E}_{n+1}. \end{aligned} \quad (14)$$

Alternatively,

$$\begin{aligned} \mathbf{e}_{n+2} + [-\mathbf{K} - \mathbf{I}] \mathbf{e}_{n+1} + \mathbf{k}_1 \mathbf{e}_{n+1} \Delta t + \mathbf{K} \mathbf{e}_n \\ = -(\mathbf{h}_{n+1} - \mathbf{h}_n) - (\mathbf{E}_{n+1} - \mathbf{E}_n). \end{aligned} \quad (15)$$

Therefore, \mathbf{K} and \mathbf{k}_1 are chosen in order to ensure the stability of the linear system represented in the left-hand side of (15). To achieve this condition, the roots of such polynomial (r_i) must lie inside the unit circle. Then, $\mathbf{e}_n \rightarrow 0$ when $n \rightarrow \infty$; that is, the error tends to zero as long as uncertainties are constant.

Once the integral of the error has been added, the control action is calculated following the same design procedure based on linear algebra presented in [30]. Therefore, the new control law u_n can be obtained through the least squares procedure as

$$\begin{aligned} u_{1n} &= \frac{P_n V_n (\chi (T_n - P_n) - (P_{\text{ref},n+1} - k_P (P_{\text{ref},n} - P_n) - P_n + k_{P1} U_{P1,n+1}) / \Delta t) + T_n V_n (\psi X_n - (T_{\text{ref},n+1} - k_T (T_{\text{ref},n} - T_n) - T_n + k_{T1} U_{T1,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2} \\ &+ \dots \\ &+ \frac{X_n V_n (\mu X_n - (X_{\text{ref},n+1} - k_X (X_{\text{ref},n} - X_n) - X_n + k_{X1} U_{X1,n+1}) / \Delta t) + (20 - S_n) V_n (7.3 \mu X_n + (S_{ez,n+1} - k_S (S_{ez,n} - S_n) - S_n + k_{S1} U_{S1,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2}. \end{aligned} \quad (16)$$

2.2. Multiple Integral Actions. Assume that the uncertainty can be modeled by a function with a second-order difference equal to zero; that is, $\delta^2 \mathbf{E}_n = \delta(\delta \mathbf{E}_n) = \delta(\mathbf{E}_{n+1} - \mathbf{E}_n) = \mathbf{E}_{n+2} - 2\mathbf{E}_{n+1} + \mathbf{E}_n = \mathbf{0}$. Similar to (10), a double integrator is now introduced, and new variables \mathbf{U}_1 and \mathbf{U}_2 are defined:

$$\mathbf{U}_{2,n+1} = \mathbf{U}_{2,n} + \int_{n\Delta t}^{(n+1)\Delta t} \mathbf{U}_1(t) dt \cong \mathbf{U}_{2,n} + \Delta t \mathbf{U}_{1,n+1}. \quad (17)$$

In this case, the control action (A.7) will include an additional term in (A.5) as follows:

$$\begin{aligned} \mathbf{z}_{n+1} &= \mathbf{z}_{\text{ref},n+1} - \mathbf{K}(\mathbf{z}_{\text{ref},n} - \mathbf{z}_n) + \mathbf{k}_1 \mathbf{U}_{1,n+1} + \mathbf{k}_2 \mathbf{U}_{2,n+1}, \\ \mathbf{k}_2 &= \begin{bmatrix} k_{P2} & 0 & 0 & 0 \\ 0 & k_{T2} & 0 & 0 \\ 0 & 0 & k_{X2} & 0 \\ 0 & 0 & 0 & k_{S2} \end{bmatrix}, \end{aligned} \quad (18)$$

where \mathbf{K} , \mathbf{k}_1 , and \mathbf{k}_2 are the proportional, integral, and double-integral design parameters, respectively. With a similar procedure to that used for deriving (15) and after adding

two integrators ($\mathbf{k}_1 \mathbf{U}_{1,n+1}$ and $\mathbf{k}_2 \mathbf{U}_{2,n+1}$), the error dynamics can be expressed as

$$\begin{aligned} \mathbf{e}_{n+3} + [-\mathbf{K} - 2\mathbf{I}] \mathbf{e}_{n+2} + (\mathbf{k}_1 + \mathbf{k}_2 \Delta t) \mathbf{e}_{n+2} \Delta t \\ + [2\mathbf{K} + \mathbf{I}] \mathbf{e}_{n+1} - \mathbf{k}_1 \mathbf{e}_{n+1} \Delta t - \mathbf{K} \mathbf{e}_n \\ = \begin{bmatrix} h_{P,n+2} - 2h_{P,n+1} + h_{P,n} \\ h_{T,n+2} - 2h_{T,n+1} + h_{T,n} \\ h_{X,n+2} - 2h_{X,n+1} + h_{X,n} \\ 0 \end{bmatrix} - \delta^2 \mathbf{E}_n. \end{aligned} \quad (19)$$

As suggested by (19), under the assumption of a constant or linear varying uncertainty ($\delta^2 \mathbf{E}_n = \mathbf{0}$), the uncertainty has no influence on the error dynamics. As in the case of the single integrator, the controller parameters \mathbf{K} , \mathbf{k}_1 , and \mathbf{k}_2 should be chosen in order to ensure the stability of the left-hand side of (19).

The previous derivation can be generalized for uncertainties that can be approximated with a $q - 1$ -order polynomial; and, therefore, the influence of \mathbf{E}_n upon \mathbf{e}_n would be cancelled off by addition of q integrators.

The linear algebra-based procedure enables the calculation of the following control law:

$$\begin{aligned}
u_{2n} = & \frac{P_n V_n (\chi (T_n - P_n) - (P_{\text{ref},n+1} - k_P (P_{\text{ref},n} - P_n) - P_n + k_{P1} U_{P1,n+1} + k_{P2} U_{P2,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2} + \dots \\
& + \frac{T_n V_n (\psi X_n - (T_{\text{ref},n+1} - k_T (T_{\text{ref},n} - T_n) - T_n + k_{T1} U_{T1,n+1} + k_{T2} U_{T2,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2} + \dots \\
& + \frac{X_n V_n (\mu X_n - (X_{\text{ref},n+1} - k_X (X_{\text{ref},n} - X_n) - X_n + k_{X1} U_{X1,n+1} + k_{X2} U_{X2,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2} + \dots \\
& + \frac{(20 - S_n) V_n (7.3 \mu X_n + (S_{\text{ez},n+1} - k_S (S_{\text{ez},n} - S_n) - S_n + k_{S1} U_{S1,n+1} + k_{S2} U_{S2,n+1}) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2}.
\end{aligned} \tag{20}$$

3. Neural State Estimator

The proposed feedback control technique requires reliable estimates of several state variables. The state estimation error, $\tilde{\mathbf{x}}_n$, is defined as

$$\tilde{\mathbf{x}}_n = \mathbf{x}_n - \hat{\mathbf{x}}_n, \tag{21}$$

where $\hat{\mathbf{x}}_n$ is the vector of estimated state variables and $\mathbf{x}_n = [\hat{P}_n, \hat{T}_n, \hat{X}_n]$ is the vector of online measured states. The following neural estimator represents the nonlinear dynamic of the bioreactor described in (7):

$$\begin{aligned}
\mathbf{x}_{n+1} &= \boldsymbol{\theta}^{*T} \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) + \boldsymbol{\varepsilon}_n, \\
\mathbf{x}_0 &= \mathbf{x}(0),
\end{aligned} \tag{22}$$

where $\mathbf{v}_n = [u_n, S_n, V_n]$ is the input vector to the neural estimator, $\boldsymbol{\theta}^*$ is the optimal weight vector, $\boldsymbol{\varepsilon}$ is the neural approximation error, ξ_i is the RBF that represents each neuron in the hidden layers, subindex i indicates the neuron number of the RBF, and m is the maximum number of neurons. Every neuron of the hidden layer is modeled as

$$\begin{aligned}
\xi_i(\mathbf{x}_n, \mathbf{v}_n) &= \exp\left(-\frac{1}{2\sigma_i^2} \left\| \begin{bmatrix} \mathbf{x}_n \\ \mathbf{v}_n \end{bmatrix} - \mathbf{c}_i \right\|^2\right), \\
i &= 1, 2, \dots, m,
\end{aligned} \tag{23}$$

where \mathbf{c}_i and σ_i are two parameter vectors that represent the centers and widths of the RBF, respectively.

In the bioreactor under study, biomass (X) cannot be measured online and the protein concentrations (P, T) cannot be measured at all; then a state estimator function based on (22) is determined as follows:

$$\hat{\mathbf{x}}_{n+1} = \hat{\boldsymbol{\theta}}_n^T \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n). \tag{24}$$

From the difference between (22) and (24), the neural identification error can be described by

$$\begin{aligned}
\tilde{\mathbf{x}}_{n+1} &= \mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1} = \boldsymbol{\theta}^{*T} \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) - \hat{\boldsymbol{\theta}}_n^T \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) + \boldsymbol{\varepsilon}_n \\
&= \tilde{\boldsymbol{\theta}}_n^T \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) + \boldsymbol{\varepsilon}_n,
\end{aligned} \tag{25}$$

where $\tilde{\boldsymbol{\theta}}_n$ represents the neural weight vector estimation error and it is defined as

$$\tilde{\boldsymbol{\theta}}_n = \boldsymbol{\theta}^* - \boldsymbol{\theta}_n. \tag{26}$$

In order to train the neural network for identification, an offline data set of (\mathbf{x}, \mathbf{u}) will be used. This training data will be obtained after following the same procedure presented in [39] and then applied in [40, 41]. A similar procedure was also used in [42]. The learning rule utilized to train the neural network is demonstrated in the next theorem.

Theorem 3. *The model of the fed-batch bioreactor (see (A.1) and (2)) can be approximated by the neural network (24) using a neuronal adjustment law defined by*

$$\Delta \tilde{\boldsymbol{\theta}}_n = -\gamma \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) \tilde{\mathbf{x}}_n^T \boldsymbol{\Lambda}. \tag{27}$$

Proof. The demonstration of this theorem is presented in Appendix B. \square

As recommended by [35], the estimator was evaluated to test its performance and avoid biased estimates. Table 1 summarizes the implemented tests, and their results were evaluated through the average values of the mean squared errors (MSE) corresponding to P, T , and X . The obtained results show an acceptable performance of the RBF neural network for estimating the nonmeasurable states. Note that none of the MSE values exceeds the thresholds proposed by [43], 1×10^{-3} . Figure 1 shows the proposed closed-loop architecture. Note that the controller requires information on both measured and estimated variables.

4. Results and Discussion

All simulation examples are based on the bioprocess originally modeled by [31]. The available optimal profiles for such process are also adopted [32]. Equation (A.1) represents the fed-batch bioreactor model. The effectiveness of the new control law is tested through some simulation examples. First, the optimal controller parameters are tuned through a Monte Carlo experiment. Then, a sinusoidal perturbation is added to some model parameters. Also, the controller performance

TABLE I: Performance evaluation of the RBF neural network.

Test	MSE _T	MSE _p	MSE _X
Normal operating conditions	1.58×10^{-6}	2.14×10^{-6}	7.37×10^{-4}
Continuous variations in the substrate concentration (15%)	2.00×10^{-6}	2.47×10^{-6}	8.17×10^{-4}
20% of error in model parameters (S_F and $Y_{S/X}$)	1.98×10^{-6}	2.70×10^{-6}	9.51×10^{-4}
20% lesser in the initial nominal operating conditions in V , S , and X	1.83×10^{-6}	2.25×10^{-6}	3.80×10^{-4}
Normally distributed noise with zero mean and variance of 0.2 in u and S	1.93×10^{-6}	2.35×10^{-6}	7.51×10^{-4}

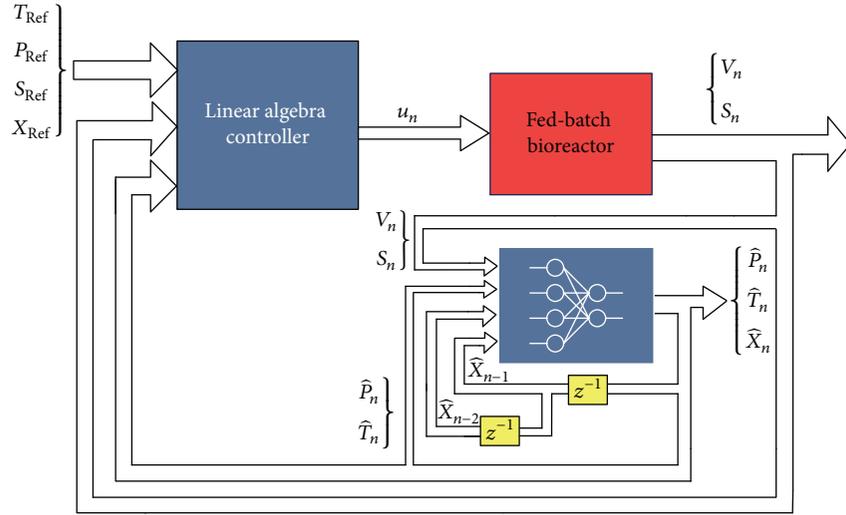


FIGURE 1: The proposed closed-loop control architecture: the linear algebra-based controller and the Radial Basis Function (RBF) neural network estimator.

under uncertain time-varying parameters is verified through a Monte Carlo Randomized Algorithm. In other examples, the functionality of the specific growth rate is changed. Finally, some parameters are contaminated by additive white noise.

As previously discussed, the behavior of the controlled system depends on parameters $\mathbf{k}, \mathbf{K}_1, \dots, \mathbf{K}_p$, where p is the number of integrators. For $p = 0, 1, 2$, the controller parameters are determined through a Monte Carlo method, where the number of simulations is determined by application of Theorem 2 of [30]. All simulations are implemented in Matlab™ (MathWorks, Inc.), with a sample time $\Delta t = 0.01$ h. For each controller (i.e., for each p), 1,000 simulations are run; and the controller parameters are randomly chosen, such that the roots (r_i) of the linear systems defined in the right-hand side of (9a) and (9b) and left-hand side of (15) and (19) verify $r_i \in (0, 1)$. More specifically, for $p = 0$, the controller parameters are directly taken from [30]. For $p = 1$, 1,000 new simulations are performed, and each controller parameter is chosen according to

$$\begin{aligned} k^i &= r_1 r_2, \\ K_1^i &= \frac{-r_1 - r_2 + r_1 r_2 + 1}{\Delta t}, \end{aligned} \quad (28)$$

where the superscript i refers to each state variable and r_1 and r_2 are the roots of the linear system in the left-hand side of (15). For $p = 2$, 1,000 simulations are performed, and each controller parameter is chosen according to

$$\begin{aligned} k^i &= r_1 r_2 r_3, \\ K_1^i &= \frac{2r_1 r_2 r_3 + 1 - r_1 r_2 - r_1 r_3 - r_2 r_3}{\Delta t}, \\ K_2^i &= \frac{-r_1 - r_2 - r_3 - r_1 r_2 r_3 + 1 + r_1 r_2 + r_1 r_3 + r_2 r_3}{\Delta t^2}, \end{aligned} \quad (29)$$

where r_1 , r_2 , and r_3 are the roots of the linear system in (19).

Table 2 summarizes the results obtained with each controller. For comparison purposes, the performance of the proposed strategy is compared with that provided by a Parallel Two-Degree-of-Freedom PID Controller (2-DOF PID), which computes a weighted difference signal for each of the proportional, integral, and derivative actions according to the set point weights specified by the user [44]. In this case, the culture cell density has been chosen as the controlled variable. The 2-DOF PID was tuned with the automatic tuning of the Simulink™ Control Design software that provides the proportional, integral, and derivative gains, as well as the filter coefficient.

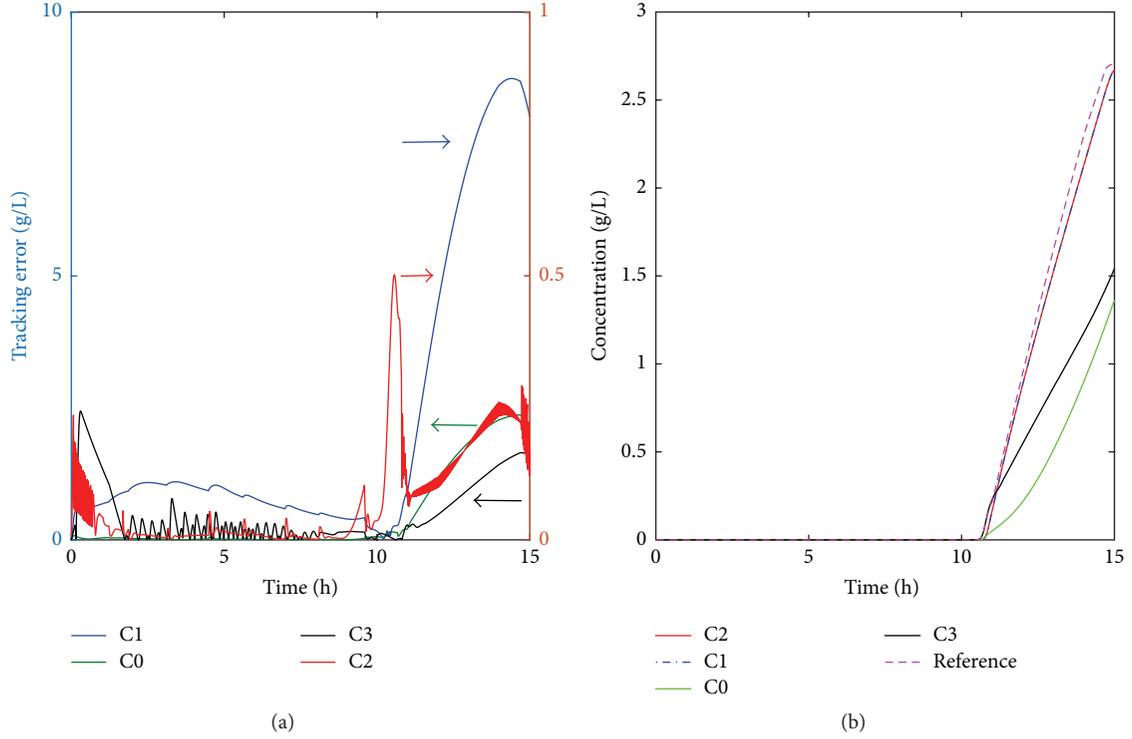


FIGURE 2: (a) Comparison of the tracking errors (see (31)) (left ordinate axis: C0 and C3; right axis: C2 and C1). (b) Total protein concentration profiles: C1 and C2 successfully reach the maximum production of protein in the presence of time-varying parameters.

TABLE 2: Main simulation results in the Monte Carlo experiments.

Controller	Controller parameters		
$p = 0$	$K_p = 0.23$		
	$K_T = 0.14$		
	$K_X = 0.52$		
	$K_S = 0.23$		
$p = 1$	$K_p = 0.7758$	$k_{p1} = 0.043$	
	$K_T = 0.6181$	$k_{T1} = 0.083$	
	$K_X = 0.8173$	$k_{X1} = 0.008$	
	$K_S = 0.0356$	$ks_1 = 0.013$	
$p = 2$	$K_p = 0.7758$	$k_{p1} = 0.053$	$k_{p2} = 0.037$
	$K_T = 0.6181$	$k_{T1} = 0.084$	$k_{T2} = 0.052$
	$K_X = 0.8173$	$k_{X1} = 0.097$	$k_{X2} = 0.065$
	$K_S = 0.0356$	$ks_1 = 0.099$	$ks_2 = 0.015$

4.1. *Time-Varying Parameters and Input Disturbances.* As in [18, 45], the following expressions are adopted for simulating uncertainties in the parameters and the feed flow rate:

$$\begin{aligned}
 \hat{Y}_{S/X} &= 7.3 + 0.05 \sin(t), \\
 \hat{S}_F &= 20 + 0.05 \sin(t), \\
 \hat{u} &= u + 0.05 \sin(t).
 \end{aligned} \tag{30}$$

The tracking error (e_n) is defined as

$$\begin{aligned}
 \|e_n\|_2 &= \sqrt{(P_{\text{ref},n} - P_n)^2 + (T_{\text{ref},n} - T_n)^2 + (X_{\text{ref},n} - X_n)^2 + (S_{e_z,n} - S_n)^2}.
 \end{aligned} \tag{31}$$

In what follows, we shall call C0, C1, and C2 the controllers that use $p = 0$, $p = 1$, and $p = 2$ integrators, respectively, and C3 the 2-DOF PID controller. Particularly note that C0 corresponds to the controller designed in [30]. In spite of the assumed time-varying parameters, Figure 2(a) shows that C2 produces the minimum tracking error. According to Figure 2(b), both C2 and C1 are effective to reach the desired protein concentration profile, while C0 and C3 produce large errors.

In several research fields, probabilistic methods have been found to be useful for dealing with problems related to robustness of systems affected by uncertainties [46]. Particularly, the Monte Carlo Randomized Algorithm has been used in many applications such as radioactive decay, power system generation, and traffic on roads [47]. In the control area, Monte Carlo methods are effective tools for the analysis of probabilistically robust control schemes [30, 46]. A Monte Carlo simulation is here performed to demonstrate the effectiveness of C2 from a statistical point of view [48–50], when uncertainties are introduced by time-varying parameter. To this effect, 1,000 random values of S_F and $Y_{S/X}$ are chosen in a range of $\pm 25\%$ of their nominal values. Figures 3(a), 3(c), 3(e), and 3(g) show that only C2 and C1 were able to adequately track the reference profile. In fact, both controller

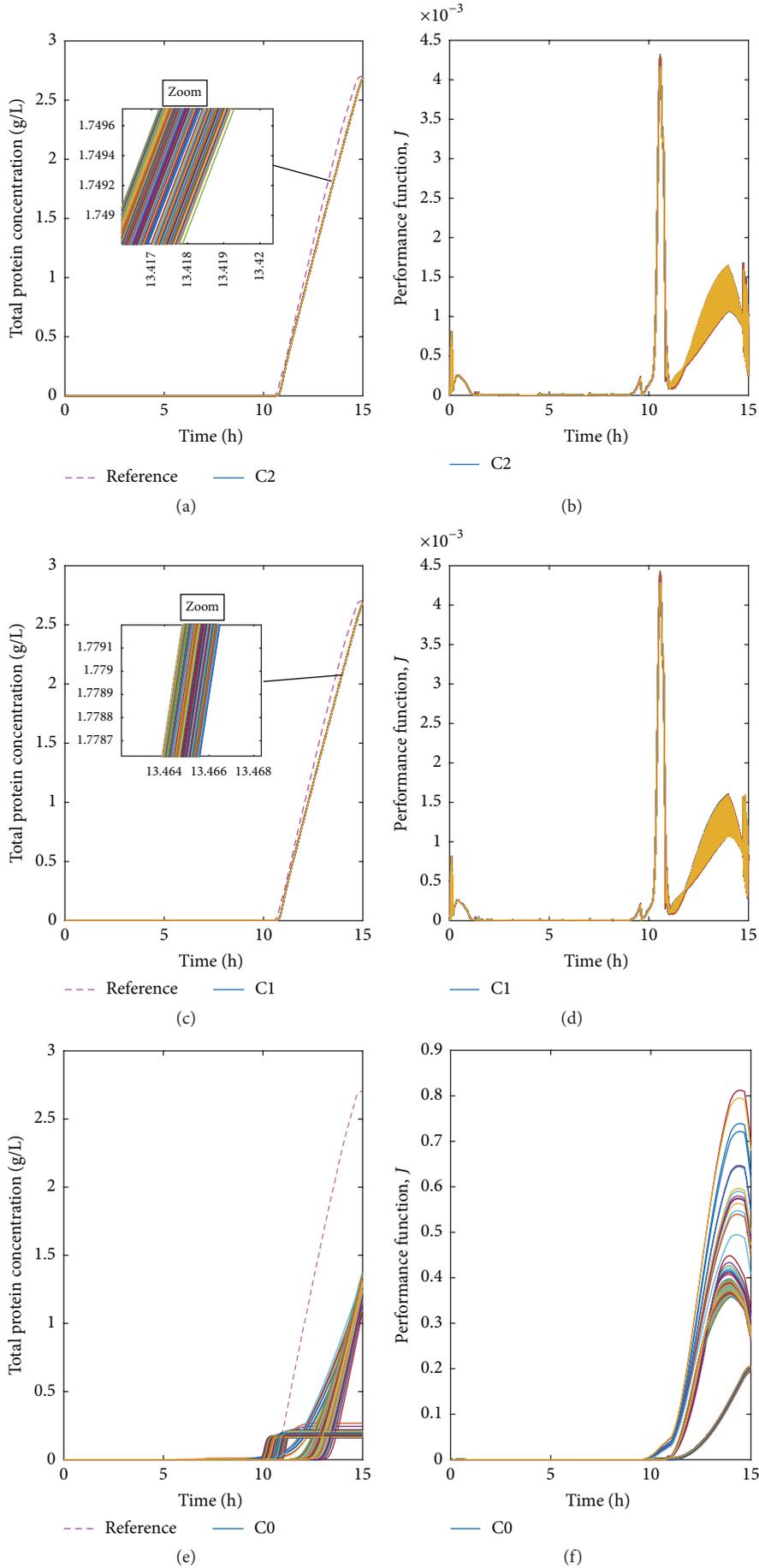


FIGURE 3: Continued.

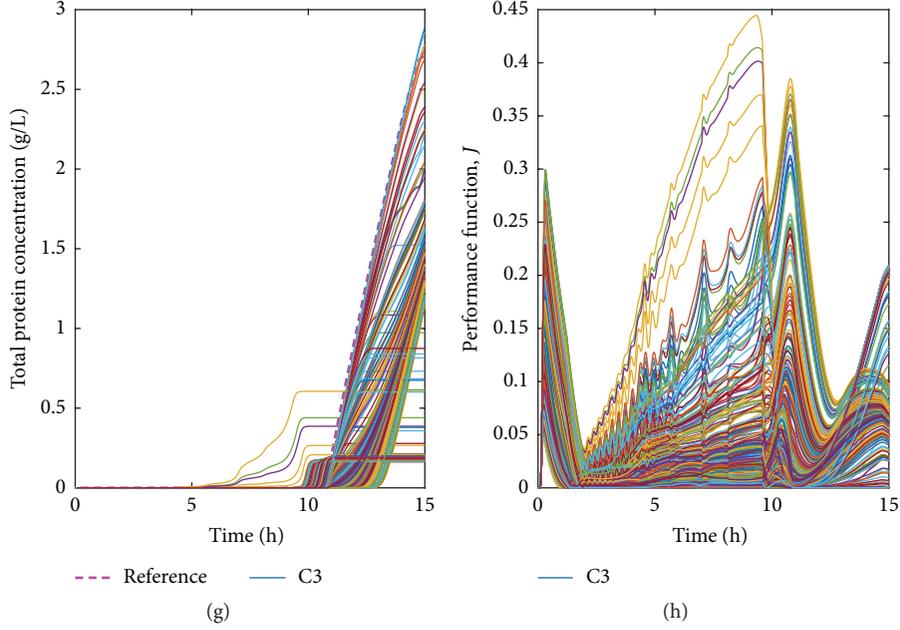


FIGURE 3: Total protein concentration ((a), (c), (e), and (g)) and performance index J ((b), (d), (f), and (h)) for 1,000 random values of S_F and $Y_{S/X}$ (controllers are indicated in each figure).

responses are quite similar, thus suggesting that the time-varying parameter has not affected their performances. For

comparison, the tracking error was evaluated through the following performance index J :

$$J = \frac{0.25 \sum_{n=1}^{N_{\text{ref}}} ((P_{\text{ref},n} - P_n)^2 + (T_{\text{ref},n} - T_n)^2 + (X_{\text{ref},n} - X_n)^2 + (S_{\text{ez},n} - S_n)^2)}{\Delta t N_{\text{ref}}}, \quad (32)$$

where N_{ref} represents the number of points of the reference profile. Figures 3(b), 3(d), 3(f), and 3(h) confirm a better performance of C2 and C1 over C0 and C3. In fact, J is quite small for C2 and C1, even though 25% of uncertainty in the main bioreactor parameters was adopted. Since 1,000 simulations with random values of S_F and $Y_{S/X}$ were carried out and the tracking errors remained bounded, the performance of C2 and C1 will be acceptable with 99% of probability whatever the parameter values are in a range of $\pm 25\%$ [46].

4.2. Uncertain Nonlinearities. Uncertain nonlinearities are often present in many bioprocesses. For instance, the specific growth rate, μ , can suffer from important changes from batch to batch, and unfortunately such variability is impossible to be accurately modeled. There are a number of models in the literature for μ [8], for example, the third expression of (A.2), for the bioprocess currently under study [31]. To demonstrate the controller performance, an additional term can be included in the calculation of μ as follows [18, 51]:

$$\hat{\mu} = \frac{(21.87 + 0.28 \sin(t)) S}{(S + 0.4)(S + 62.5)}. \quad (33)$$

Figure 4(a) compares the performances reached by C2, C1, C0, and C3 under the simulated uncertain nonlinearities in

μ . The lower tracking errors correspond to C2. Figure 4(b) shows that only C2 and C1 are able to reach the maximum concentration of the total protein.

4.3. Unmodeled Dynamics. The presence of white noise in μ_{max} leads to a mathematical model governed by a set of stochastic differential equations. In order to properly deal with this type of uncertainties, a kind of stochastic control approach should be used. As an alternative to the complex theory of stochastic control, a simpler deterministic strategy is here proposed. To this effect, note first that all parameters in (A.1) and (A.2) are truly affected by environmental noise, and consequently they always fluctuate at long times around some average values. For simplicity, we will consider that only μ_{max} is affected by randomness and changes according to the following expression [52, 53]:

$$d\mu_{\text{max}} = \mu_{\text{max}} + \alpha dB(t), \quad (34)$$

where $dB(t)$ is a standard Brownian motion defined on a complete probability space and $\alpha \geq 0$ represents the intensity of the noisy signal.

A white noise can be approximated through a band-limited white noise (BLWN), which is simply implemented through a proper block of Simulink. Theoretically, a white

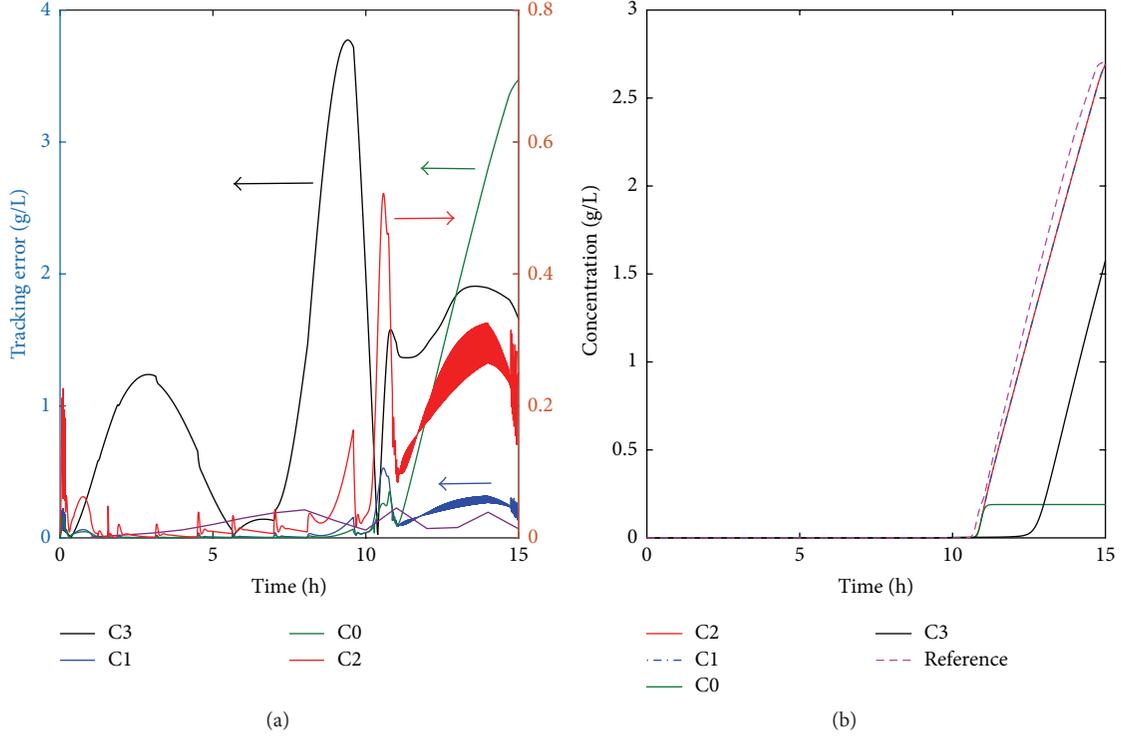


FIGURE 4: (a) Comparison of the tracking errors (see (31)) under uncertain nonlinearities in μ (left ordinate axis: C0, C1, and C3; right axis: C2). (b) Total protein concentration profiles.

noise is uncorrelated; it exhibits a flat power spectral density, and therefore it is characterized by an infinite energy. The BLWN block can simulate the effect of white noise by using a random sequence with a correlation time much smaller than the shortest time constant of the system. The correlation time of the noise is the sample rate of the block [54] and can be estimated as

$$t_c \approx \frac{2\pi}{100} \frac{1}{f_{\max}} = \frac{2\pi}{100} \frac{1}{0.5/\Delta t_{\max}}, \quad (35)$$

where t_c is the correlation time, f_{\max} is the bandwidth of the system, and Δt_{\max} is the maximum sample time for which the system is stable. For the feedback control presented in this work, $\Delta t_{\max} = 0.0145$ h. Then, from (35), $t_c = 0.0018$ h.

Many researchers have utilized this methodology to prove the robustness of the bioreactors controllers [19, 22–24, 55]. In the current simulations, a BLWN with a noise power of 0.9 and $\alpha = 0.5$ is introduced along with the feedback signal. Thus, the maximum specific growth rate affected with BLWN is calculated as

$$\tilde{\mu}_{\max} = \mu_{\max} + 0.05\text{BLWN}. \quad (36)$$

Figure 5(a) indicates that C2 and C1 adequately work in the presence of unmodeled dynamics. In fact, the tracking errors are quite small in comparison with C0 and C3. Figure 5(b) shows the profile tracking for each controller. Again, C2 and C1 are able to follow the reference profile, while C0 and C3 fail to reach the maximum protein concentration.

5. Conclusions

A new control law for tracking an optimal concentration profile in a fed-batch bioreactor has been presented. The proposed method allows controlling a nonlinear system in presence of unavoidable uncertainties in bioreactors: time-varying parameters, uncertain nonlinearities, and unmodeled dynamics.

In order to deal with these uncertainties, some integrators have been added to the original controller proposed in [30], and a Radial Basis Function (RBF) neural network was employed to estimate X , P , and T . The integrators have been chosen based on the variation hypothesis of the modeling error (E_n): if a q -grade polynomial can approximate E_n , then $q + 1$ integrators will be added in order to avoid its influence on the tracking error (e_n).

Several tests were carried out to demonstrate the effectiveness of the proposed methodology. The controllers are tuned through a Monte Carlo experiment and under stability conditions. To evaluate the performance of the designed controllers, different uncertainties were simulated. It was proven that the proposed methodology successfully compensates the uncertainties, and the tracking errors decrease as long as the number of integrators increases. The performance of the proposed controller with two integrators (C2) is remarkable, and the complexity of the control algorithm is relatively small. Moreover, the addition of integrators does not increase significantly the calculation time.

In comparison to previous strategies that deal with similar control problems (e.g., [17, 21, 22, 24, 25, 27]), the proposed

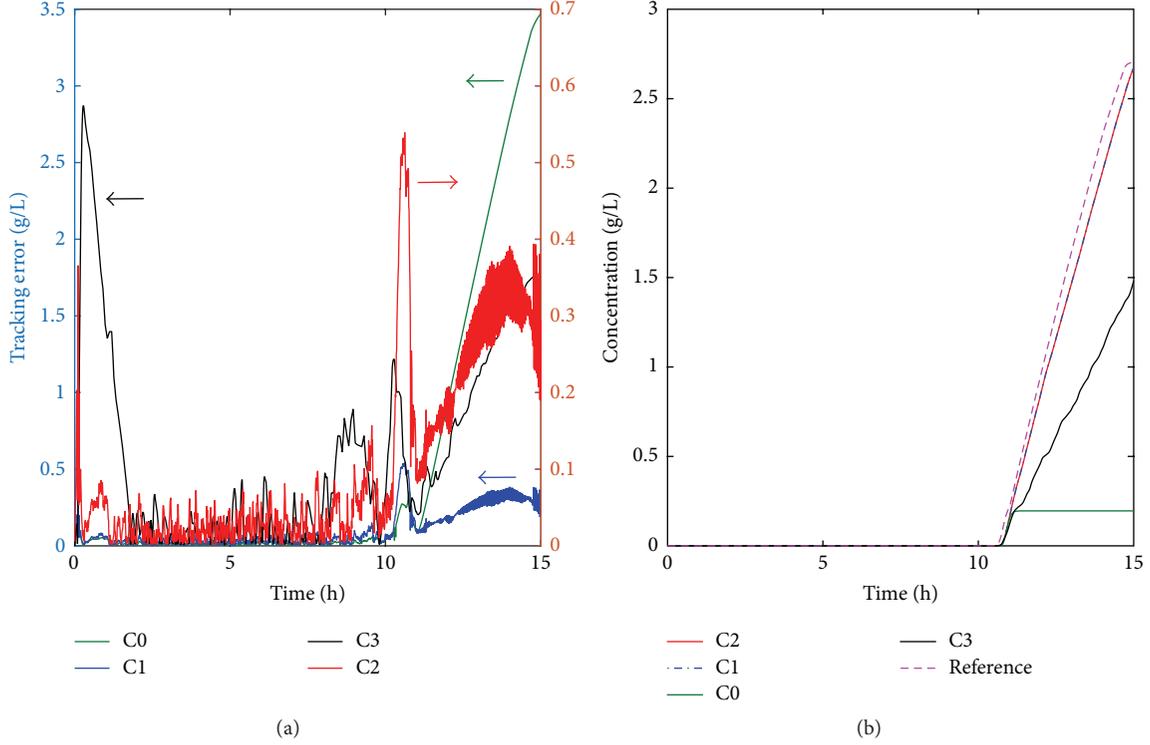


FIGURE 5: (a) Comparison of the tracking errors (see (31)) (right ordinate axis: C2; left axis: C0, C1, and C3). (b) Total protein concentration profiles.

controllers present the advantage of avoiding a stochastic modeling, as it is typically required to deal with systems where the parameters are contaminated by white noise. The trade-off between the response speed and the stability is solved through a simple Monte Carlo experiment. In addition, the controller design is mathematically simple. Currently, the developed methodology for the controller design is being applied to another bioprocess. Besides, the inclusion of saturation of the control signals and sliding mode control schemes in the formulation of the problem will be addressed in future contributions.

Appendix

A. Linear Algebra-Based Controller

Below is a brief description of the fed-batch bioreactor and the controller derived in [30]. The state variables in a fed-batch reactor can be described by the following set of equations proposed in [31]:

$$\begin{aligned}
 \frac{dP}{dt} &= \chi(T - P) - \frac{uP}{V}, \\
 \frac{dT}{dt} &= \psi X - \frac{uT}{V}, \\
 \frac{dX}{dt} &= \mu X - \frac{uX}{V}, \\
 \frac{dS}{dt} &= -Y_{S/X}\mu X + \frac{u(S_F - S)}{V}
 \end{aligned} \tag{A.1}$$

with

$$\begin{aligned}
 \chi &= \frac{4.75\mu}{0.12 + \mu}, \\
 \psi &= \frac{Se^{-5S}}{0.1 + S}, \\
 \mu &= \frac{\mu_{\max}S}{(S + 0.4)(S + 62.5)}, \\
 u &= \frac{dV}{dt}.
 \end{aligned} \tag{A.2}$$

In (A.1) and (A.2), the following variables and parameters are used: the amount of secreted protein per culture volume unit (P), the total protein amount per culture volume unit (T), the culture cell density (X), the culture glucose concentration (S), and the culture volume (V); u is the feed flow rate, S_F is the glucose concentration of the feed stream, $Y_{S/X}$ is the yield of glucose per cell mass, and ψ , μ , and χ are the protein expression rate, the specific growth rate of the host cell, and the protein secretion rate, respectively. The optimal profiles for the state variables calculated in [32] were chosen as the reference profiles $[P_{\text{ref}}, T_{\text{ref}}, X_{\text{ref}}, S_{\text{ref}}]^T$ to be tracked by the control algorithm. The nominal values of the model parameters are $S_F = 20 \text{ gL}^{-1}$ and $Y_{S/X} = 7.3$. Initial conditions for the state variables are $P_0 = 0 \text{ gL}^{-1}$, $T_0 = 0 \text{ gL}^{-1}$, $X_0 = 1 \text{ gL}^{-1}$, $S_0 = 5 \text{ gL}^{-1}$, and $V_0 = 1 \text{ L}$.

In order to follow the reference profiles, the required control action is calculated from the process model. First,

(A.1) can be rewritten in a discrete version using the Euler approximation:

$$\begin{bmatrix} \frac{P_n}{V_n} \\ \frac{T_n}{V_n} \\ \frac{X_n}{V_n} \\ \frac{S_F - S_n}{V_n} \end{bmatrix} u_n = \begin{bmatrix} \chi_n (T_n - P_n) - \frac{P_{n+1} - P_n}{\Delta t} \\ \psi_n X_n - \frac{T_{n+1} - T_n}{\Delta t} \\ \mu_n X_n - \frac{X_{n+1} - X_n}{\Delta t} \\ 7.3\mu_n X_n + \frac{S_{n+1} - S_n}{\Delta t} \end{bmatrix}. \quad (\text{A.3})$$

Let us denote by \mathbf{z}_n the state vector and by \mathbf{c}_n and \mathbf{d}_n the input vectors. Then, the state vector in the sample time $(n + 1)$ can be expressed as

$$\begin{aligned} \mathbf{z}_{n+1} &= \mathbf{z}_n + \mathbf{c}_n \Delta t - \mathbf{d}_n u_n \Delta t, \\ \mathbf{z}_n &= \begin{bmatrix} P_n \\ T_n \\ X_n \\ S_n \end{bmatrix}; \\ \mathbf{c}_n &= \begin{bmatrix} \chi(T_n - P_n) \\ \psi X_n \\ \mu X_n \\ -7.3\mu X_n \end{bmatrix}; \\ \mathbf{d}_n &= \begin{bmatrix} \frac{P_n}{V_n} & \frac{T_n}{V_n} & \frac{X_n}{V_n} & \frac{S_n - 20}{V_n} \end{bmatrix}^T. \end{aligned} \quad (\text{A.4})$$

The following replacements should be done in (A.3):

$$\mathbf{z}_{n+1} = \mathbf{z}_{\text{ref},n+1} - \mathbf{K}(\mathbf{z}_{\text{ref},n} - \mathbf{z}_n),$$

$$\begin{aligned} u_n &= \frac{P_n V_n (\chi(T_n - P_n) - (P_{\text{ref},n+1} - k_P(P_{\text{ref},n} - P_n) - P_n) / \Delta t) + T_n V_n (\psi X_n - (T_{\text{ref},n+1} - k_T(T_{\text{ref},n} - T_n) - T_n) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2} \\ &+ \dots \\ &+ \frac{X_n V_n (\mu X_n - (X_{\text{ref},n+1} - k_X(X_{\text{ref},n} - X_n) - X_n) / \Delta t) + (20 - S_n) V_n (7.3\mu X_n + (S_{\text{ez},n+1} - k_S(S_{\text{ez},n} - S_n) - S_n) / \Delta t)}{P_n^2 + T_n^2 + X_n^2 + (20 - S_n)^2}. \end{aligned} \quad (\text{A.7})$$

B. Proof of Theorem 3

Assumption B.1. The optimal approximation error ε is bounded by a constant: $\|\varepsilon\| \leq \bar{\varepsilon}$.

Proof of Theorem 3. Define a candidate function of Lyapunov, with V_m being a positive definite function given by

$$V_m = \tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + \gamma^{-1} \text{tr}(\tilde{\boldsymbol{\theta}}_{n-1}^T \tilde{\boldsymbol{\theta}}_{n-1}). \quad (\text{B.1})$$

$$\mathbf{K} = \begin{bmatrix} k_P & 0 & 0 & 0 \\ 0 & k_T & 0 & 0 \\ 0 & 0 & k_X & 0 \\ 0 & 0 & 0 & k_S \end{bmatrix}, \quad (\text{A.5})$$

where \mathbf{K} is the controller parameter matrix and $k_P, k_T, k_X, k_S \in (0, 1)$. The Monte Carlo Randomized Algorithm was applied to select the controller parameter matrix. This procedure does not ensure reaching the global optimum. Nevertheless, based on Theorem 2 of [30], a large number of trials were made in order to guarantee a large confidence interval. Then, (A.3) can be rewritten as

$$\begin{bmatrix} \frac{P_n}{V_n} \\ \frac{T_n}{V_n} \\ \frac{X_n}{V_n} \\ \frac{20 - S_n}{V_n} \end{bmatrix} u_n = \begin{bmatrix} \chi(T_n - P_n) - \frac{P_{\text{ref},n} - k_P(P_{\text{ref},n} - P_n) - P_n}{\Delta t} \\ \psi X_n - \frac{T_{\text{ref},n+1} - k_T(T_{\text{ref},n} - T_n) - T_n}{\Delta t} \\ \mu X_n - \frac{X_{\text{ref},n+1} - k_X(X_{\text{ref},n} - X_n) - X_n}{\Delta t} \\ 7.3\mu X_n + \frac{S_{\text{ref},n+1} - k_S(S_{\text{ref},n} - S_n) - S_n}{\Delta t} \end{bmatrix}. \quad (\text{A.6})$$

Equation (A.6) has an exact solution when the culture glucose concentration is adopted as a ‘‘sacrificed variable,’’ S_{ez} [30]. Then, the control action u_n can be obtained through the least squares procedure:

The difference in discrete time ΔV_m is adopted as follows:

$$\begin{aligned} \Delta V_m &= (\tilde{\mathbf{x}}_{n+1}^T \Lambda \tilde{\mathbf{x}}_{n+1} - \tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n) \\ &+ \gamma^{-1} \text{tr}(\tilde{\boldsymbol{\theta}}_n^T \tilde{\boldsymbol{\theta}}_n - \tilde{\boldsymbol{\theta}}_{n-1}^T \tilde{\boldsymbol{\theta}}_{n-1}). \end{aligned} \quad (\text{B.2})$$

And $\Delta \boldsymbol{\theta}_i$ is defined as

$$\Delta \boldsymbol{\theta}_i = \gamma_1^{-1} \text{tr}(\tilde{\boldsymbol{\theta}}_n^T \tilde{\boldsymbol{\theta}}_n - \tilde{\boldsymbol{\theta}}_{(n-1)}^T \tilde{\boldsymbol{\theta}}_{(n-1)}). \quad (\text{B.3})$$

By replacing (25) into (B.4), ΔV_m is written as

$$\begin{aligned}\Delta V_m &= (\tilde{\mathbf{x}}_{n+1}^T \Lambda \tilde{\mathbf{x}}_{n+1} - \tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n) + \Delta \boldsymbol{\theta} \\ &= ((\tilde{\mathbf{x}}_n + \Delta \tilde{\mathbf{x}}_n)^T \Lambda (\tilde{\mathbf{x}}_n + \Delta \tilde{\mathbf{x}}_n) - \tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n) + \Delta \boldsymbol{\theta} \quad (\text{B.4}) \\ &= \dots = 2\tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n + \Delta \boldsymbol{\theta}.\end{aligned}$$

By replacing (25) into (B.4), ΔV_m is written as

$$\begin{aligned}\Delta V_m &= 2\tilde{\mathbf{x}}_n^T \Lambda (\tilde{\mathbf{x}}_{n+1} - \tilde{\mathbf{x}}_n) + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n + \Delta \boldsymbol{\theta} \\ &= 2\tilde{\mathbf{x}}_n^T \Lambda (\tilde{\boldsymbol{\theta}}_n^T \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) + \boldsymbol{\varepsilon}_n - \tilde{\mathbf{x}}_n) + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n \quad (\text{B.5}) \\ &\quad + \Delta \boldsymbol{\theta}.\end{aligned}$$

From (B.3) and rewriting $\Delta \boldsymbol{\theta}$, one obtains

$$\begin{aligned}\Delta \boldsymbol{\theta} &= \gamma^{-1} \text{tr} \left(\tilde{\boldsymbol{\theta}}_n^T \tilde{\boldsymbol{\theta}}_n - [\tilde{\boldsymbol{\theta}}_n - \Delta \tilde{\boldsymbol{\theta}}_n]^T [\tilde{\boldsymbol{\theta}}_n - \Delta \tilde{\boldsymbol{\theta}}_n] \right), \quad (\text{B.6}) \\ \Delta \boldsymbol{\theta} &= 2\gamma^{-1} \text{tr} \left(\tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right) - \gamma^{-1} \text{tr} \left(\Delta \tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right).\end{aligned}$$

Then, (B.6) is replaced into (B.5) and, after reorganizing some terms, one obtains

$$\begin{aligned}\Delta V_m &= -2\tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \tilde{\boldsymbol{\theta}}_n^T \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{u}_n) + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n \\ &\quad + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n + \Delta \boldsymbol{\theta} \\ &= -2\tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n \quad (\text{B.7}) \\ &\quad + \text{tr} \left(2\tilde{\boldsymbol{\theta}}_n^T [\boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) \tilde{\mathbf{x}}_n^T \Lambda + \gamma^{-1} \Delta \tilde{\boldsymbol{\theta}}_n] \right) \\ &\quad - \gamma^{-1} \text{tr} \left(\Delta \tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right).\end{aligned}$$

Replacing the adjustment laws (see (27)), $\Delta \tilde{\boldsymbol{\theta}}$, in (B.7), ΔV_m is represented by

$$\begin{aligned}\Delta V_m &= -2\tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + \Delta \tilde{\mathbf{x}}_n^T \Lambda \Delta \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n \\ &\quad - \gamma^{-1} \text{tr} \left(\Delta \tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right).\end{aligned} \quad (\text{B.8})$$

In (B.8), the increment of the model error of $\Delta \tilde{\mathbf{x}}$ is unknown, and it can be approximated by the following equation:

$$\Delta \tilde{\mathbf{x}}_{ni} = \left(\frac{\partial \tilde{\mathbf{x}}_{(n+1)i}}{\partial \tilde{\boldsymbol{\theta}}_{ni}} \right)^T \Delta \tilde{\boldsymbol{\theta}}_{ni}. \quad (\text{B.9})$$

The partial derivative of the model error depends only on the weights of the neural network and can be rewritten as

$$\Delta \tilde{\mathbf{x}}_{ni} = - \left(\frac{\partial \tilde{\mathbf{x}}_{(n+1)i}}{\partial \tilde{\boldsymbol{\theta}}_{ni}} \right)^T \Delta \tilde{\boldsymbol{\theta}}_{ni}. \quad (\text{B.10})$$

Changing the values of the weights according to (B.6) and considering that $\boldsymbol{\theta}^*$ is constant, ideal weight vectors are

required only for analytical purpose. Now reorganizing $\Delta \tilde{\boldsymbol{\theta}}_{ni}$, it can be rewritten as

$$\begin{aligned}\Delta \tilde{\boldsymbol{\theta}}_{ni} &= \tilde{\boldsymbol{\theta}}_{ni} - \tilde{\boldsymbol{\theta}}_{(n-1)i} = (\boldsymbol{\theta}_i^* - \hat{\boldsymbol{\theta}}_{ni}) - (\boldsymbol{\theta}_i^* - \hat{\boldsymbol{\theta}}_{(n-1)i}) \\ &= -\hat{\boldsymbol{\theta}}_{ni} + \hat{\boldsymbol{\theta}}_{(n-1)i} = -\Delta \hat{\boldsymbol{\theta}}_{ni}.\end{aligned} \quad (\text{B.11})$$

Equation (B.10) can be written as

$$\Delta \tilde{\mathbf{x}}_{ni} = - \left(\frac{\partial \tilde{\mathbf{x}}_{(n+1)i}}{\partial \hat{\boldsymbol{\theta}}_{ni}} \right)^T (\gamma \tilde{\mathbf{x}}_{ni} \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n)). \quad (\text{B.12})$$

Considering the value of the partial derivatives of the neural network (see (27)) and by substitution in (B.10), it yields

$$\Delta \tilde{\mathbf{x}}_{ni} = -\gamma \tilde{\mathbf{x}}_{ni} \boldsymbol{\xi}^T(\mathbf{x}_n, \mathbf{v}_n) \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n). \quad (\text{B.13})$$

The increase in the model error is defined as

$$\Delta \tilde{\mathbf{x}}_{ni} \leq \gamma |\tilde{\mathbf{x}}_{ni}|, \quad (\text{B.14})$$

where

$$\gamma = \gamma \|\boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n)\|^2, \quad (\text{B.15})$$

$$\max(\boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n)) \leq 1.$$

Consider that $\|\boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n)\|$ is a bounded function. The value γ is a learning factor of the neural network ($0 < \gamma < 1$) and it can be arbitrarily defined.

Expressing (B.14) in vectorial form,

$$\|\Delta \tilde{\mathbf{x}}_n\| \leq \gamma \|\tilde{\mathbf{x}}_n\|. \quad (\text{B.16})$$

Now, substituting the increment value of the model error in (B.16), the Lyapunov discrete difference is defined as

$$\begin{aligned}\Delta V_m &= \tilde{\mathbf{x}}_n^T \gamma^2 \Lambda \tilde{\mathbf{x}}_n - 2\tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n \\ &\quad - \gamma^{-1} \text{tr} \left(\Delta \tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right),\end{aligned} \quad (\text{B.17})$$

$$\begin{aligned}\Delta V_m &= -\tilde{\mathbf{x}}_n^T (\Lambda (1 - \gamma^2)) \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n \\ &\quad - \gamma^{-1} \text{tr} \left(\Delta \tilde{\boldsymbol{\theta}}_n^T \Delta \tilde{\boldsymbol{\theta}}_n \right).\end{aligned}$$

Now replacing the learning rule in (B.17) and applying norm yield

$$\begin{aligned}\Delta V_m &= -\tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + \gamma^2 \tilde{\mathbf{x}}_n^T \Lambda \tilde{\mathbf{x}}_n + 2\tilde{\mathbf{x}}_n^T \Lambda \boldsymbol{\varepsilon}_n \\ &\quad - \gamma^{-1} \text{tr} \left((\gamma \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) \tilde{\mathbf{x}}_n^T \Lambda)^T (\gamma \boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n) \tilde{\mathbf{x}}_n^T \Lambda) \right), \\ \Delta V_m &\leq -\|\Lambda\| \|\tilde{\mathbf{x}}_n\|^2 + \gamma^2 \|\Lambda\| \|\tilde{\mathbf{x}}_n\|^2 + 2\lambda_{\max}(\Lambda) \|\tilde{\mathbf{x}}_n\| \bar{\boldsymbol{\varepsilon}} \\ &\quad - \gamma^2 \|\Lambda\| \|\boldsymbol{\xi}(\mathbf{x}_n, \mathbf{v}_n)\|^2 \|\tilde{\mathbf{x}}_n\|^2,\end{aligned} \quad (\text{B.18})$$

$$\Delta V_m \leq \|\Lambda\| \|\tilde{\mathbf{x}}_n\|^2 + 2\|\Lambda\| \|\tilde{\mathbf{x}}_n\| \bar{\boldsymbol{\varepsilon}},$$

where $\|\xi(\mathbf{x}_n, \mathbf{v}_n)\| \leq 1$; then (B.18) can be expressed as

$$\begin{aligned} \Delta V_m &\leq -\|\Lambda\| \|\tilde{\mathbf{x}}_n\|^2 + 2\|\Lambda\| \|\tilde{\mathbf{x}}_n\| \bar{\varepsilon} \\ &= -\lambda_{\min}(\Lambda) (\|\tilde{\mathbf{x}}_n\| - 2\bar{\varepsilon}) \|\tilde{\mathbf{x}}_n\|. \end{aligned} \quad (\text{B.19})$$

From (B.19), it follows that when $\varepsilon = 0$, x_n tends to zero when n tends to infinity. If the error norm is $\|\tilde{\mathbf{x}}_n\| < 2\bar{\varepsilon}$, we can obtain that $V > 0$ and the neural weights could tend to infinity. To prevent the above situation, the next update rule is used.

$$\Delta\theta = \begin{cases} \gamma \xi(\mathbf{x}_n, \mathbf{v}_n) \tilde{\mathbf{x}}_n^T \Lambda & \text{if } \|\tilde{\mathbf{x}}_n\| \geq \bar{\mathbf{x}}_0, \|\theta\| \leq M \\ 0 & \text{somewhere else,} \end{cases} \quad (\text{B.20})$$

where $\bar{\mathbf{x}}_0$ and M are design parameters. \square

Nomenclature

Symbol

\mathbf{c}_i :	Vector of centers of each RBF function, —
dB :	Standard Brownian motion, h^{-1}
\mathbf{e}_n :	Tracking error, $\text{g}^2\text{L}^{-2}\text{h}^{-1}$
\mathbf{E} :	Additive uncertainty, gL^{-1}
f_{\max} :	System bandwidth, h^{-1}
J :	Performance index, $\text{gL}^{-1}\text{h}^{-1}$
k, K :	Controller parameters, —
M :	Neural design parameter, —
m :	Maximum number of neurons, —
N_{ref} :	Number of points of the reference profile, —
P :	Secreted protein concentration, gL^{-1}
r :	Root of a linear system, —
S :	Glucose concentration, gL^{-1}
S_F :	Glucose concentration of the feed, gL^{-1}
T :	Total protein concentration, gL^{-1}
t_c :	Correlation time, h
Δt :	Sample time, h
Δt_{\max} :	Maximum sample time, h
u :	Feed flow rate, Lh^{-1}
U :	Integral of the tracking error, g^2L^{-2}
V :	Bioreactor volume, L
V_m :	Lyapunov candidate function, —
X :	Biomass, gL^{-1}
$\tilde{\mathbf{x}}_n$:	Estimated variables vector $(\hat{P}_n, \hat{T}_n, \hat{X}_n)$, gL^{-1}
$\tilde{\mathbf{x}}_0$:	Neural design parameter, —
$Y_{S/X}$:	Yield of glucose per cell mass, —
\mathbf{z} :	State vector, gL^{-1} .

Subscripts

ez :	Sacrificed variable, —
n :	Actual sample time, —
ref:	Reference profile, —
0:	Initial condition, —

Greek Letters

α :	Intensity of the noisy signal, —
λ, δ, φ :	Random number between 0 and 1, —

Λ :	Diagonal definite positive matrix, —
$\xi(\cdot)$:	RBFs vector, —
ε_n :	Neural approximation error, —
σ_i :	Vector of widths of each RBF function, —
θ_n :	Neural weight matrix, —
θ^* :	Optimal weight matrix, —
$\tilde{\theta}_n$:	Neural error weight matrix, —
$\Delta\theta_i$:	Auxiliary vector, —
μ :	Specific growth rate, h^{-1}
μ_{\max} :	Maximum specific growth rate, h^{-1}
ψ :	Protein expression rate, —
χ :	Protein secretion rate, —
\mathbf{v}_n :	Input vector to the neural estimator $[u_n, S_n, V_n]$, —
γ :	Learning factor of the neural network, —

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was partially funded by the following Argentine institutions: National Council of Scientific and Technological Research (CONICET), Universidad Nacional de San Juan (UNSJ), and Universidad Tecnológica Nacional.

References

- [1] N. Dimitrova and M. Krastanov, "Nonlinear adaptive control of a model of an uncertain fermentation process," *International Journal of Robust and Nonlinear Control*, vol. 20, no. 9, pp. 1001–1009, 2010.
- [2] L. M. Beltrán, C. L. Garzon-Castro, F. Garcés, and M. Moreno, "Monitoring and control system used in microalgae crop," *IEEE Latin America Transactions*, vol. 10, no. 4, pp. 1993–1998, 2012.
- [3] M. Cárcamo, P. A. Saa, J. Torres et al., "Effective dissolved oxygen control strategy for high-cell-density cultures," *IEEE Latin America Transactions*, vol. 12, no. 3, pp. 389–394, 2014.
- [4] K. C. Veluvolu, Y. C. Soh, and W. Cao, "Robust discrete-time nonlinear sliding mode state estimation of uncertain nonlinear systems," *International Journal of Robust and Nonlinear Control*, vol. 17, no. 9, pp. 803–828, 2007.
- [5] S. Rómoli, A. N. Amicarelli, O. A. Ortiz, G. J. E. Scaglia, and F. di Sciascio, "Nonlinear control of the dissolved oxygen concentration integrated with a biomass estimator for production of *Bacillus thuringiensis* δ -endotoxins," *Computers and Chemical Engineering*, vol. 93, pp. 13–24, 2016.
- [6] S. Gnath, M. Jenzsch, R. Simutis, and A. Lübbert, "Control of cultivation processes for recombinant protein production: a review," *Bioprocess and Biosystems Engineering*, vol. 31, no. 1, pp. 21–39, 2008.
- [7] S. Tokat, "Sliding mode controlled bioreactor using a time-varying sliding surface," *Transactions of the Institute of Measurement & Control*, vol. 31, no. 5, pp. 435–456, 2009.
- [8] T. Egli, "Growth kinetics, bacterial," in *Encyclopedia of Microbiology*, M. Schaechter, Ed., pp. 180–193, Academic Press, Oxford, UK, 3rd edition, 2009.
- [9] S. C. Patwardhan, S. Narasimhan, P. Jagadeesan, B. Gopaluni, and S. L. Shah, "Nonlinear Bayesian state estimation: a review

- of recent developments,” *Control Engineering Practice*, vol. 20, no. 10, pp. 933–953, 2012.
- [10] H. O. Méndez-Acosta, D. U. Campos-Delgado, R. Femat, and V. González-Alvarez, “A robust feedforward/feedback control for an anaerobic digester,” *Computers and Chemical Engineering*, vol. 29, no. 7, pp. 1613–1623, 2005.
- [11] J. Alvarez-Ramírez, “Adaptive control of feedback linearizable systems: a modelling error compensation approach,” *International Journal of Robust and Nonlinear Control*, vol. 9, no. 6, pp. 361–377, 1999.
- [12] S. Tong, C. Liu, and Y. Li, “Fuzzy-adaptive decentralized output-feedback control for large-scale nonlinear systems with dynamical uncertainties,” *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 5, pp. 845–861, 2010.
- [13] Z. Chen and T. Zhang, “Dynamics of a stochastic model for continuous flow bioreactor with Contois growth rate,” *Journal of Mathematical Chemistry*, vol. 51, no. 3, pp. 1076–1091, 2013.
- [14] J. Hess and O. Bernard, “Design and study of a risk management criterion for an unstable anaerobic wastewater treatment process,” *Journal of Process Control*, vol. 18, no. 1, pp. 71–79, 2008.
- [15] G. Lara-Cisneros, R. Femat, and E. a. Pérez, “On dynamical behaviour of two-dimensional biological reactors,” *International Journal of Systems Science. Principles and Applications of Systems and Integration*, vol. 43, no. 3, pp. 526–534, 2012.
- [16] H. L. Smith and P. Waltman, *The Theory of the Chemostat: Dynamics of Microbial Competition*, vol. 13, Cambridge University Press, 1995.
- [17] A. Rincón, D. Piarpuzán, and F. Angulo, “A new adaptive controller for bio-reactors with unknown kinetics and biomass concentration: guarantees for the boundedness and convergence properties,” *Mathematics and Computers in Simulation*, vol. 112, pp. 1–13, 2015.
- [18] K. J. Keesman and V. I. Maksimov, “On feedback identification of unknown characteristics: a bioreactor case study,” *International Journal of Control*, vol. 81, no. 1, pp. 134–145, 2008.
- [19] A. E. Rodríguez, J. A. T. Muñoz, R. Luna et al., “Robust control for cultivation of microorganisms in a high density fed-batch bioreactor,” *IEEE Latin America Transactions*, vol. 13, no. 6, pp. 1927–1933, 2015.
- [20] L. Mailleret, O. Bernard, and J.-P. Steyer, “Nonlinear adaptive control for bioreactors with unknown kinetics,” *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 40, no. 8, pp. 1379–1385, 2004.
- [21] C. Bartolomeo and G. Mosè, “Type-2 fuzzy control of a bioreactor,” in *Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent System (ICIS '09)*, pp. 700–704, November 2009.
- [22] G. Lara-Cisneros, R. Femat, and D. Dochain, “An extremum seeking approach via variable-structure control for fed-batch bioreactors with uncertain growth rate,” *Journal of Process Control*, vol. 24, no. 5, pp. 663–671, 2014.
- [23] M. Guay, D. Dochain, and M. Perrier, “Adaptive extremum seeking control of continuous stirred tank bioreactors with unknown growth kinetics,” *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 40, no. 5, pp. 881–888 (2005), 2004.
- [24] C. Cimander and C.-F. Mandenius, “Bioprocess control from a multivariate process trajectory,” *Bioprocess and Biosystems Engineering*, vol. 26, no. 6, pp. 401–411, 2004.
- [25] F. Mairet and J.-L. Gouzé, “Hybrid control of a bioreactor with quantized measurements,” *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 61, no. 5, pp. 1385–1390, 2016.
- [26] F. Liping, Z. Jun, H. Xing, and H. Dong, “The design of the MISO model predictive controller for bioreactor,” *TELKOMNIKA Indonesian Journal of Electrical Engineering*, vol. 10, no. 6, 2012.
- [27] M. Elenchezhiyan and J. Prakash, “State estimation of stochastic non-linear hybrid dynamic system using an interacting multiple model algorithm,” *ISA Transactions*, vol. 58, pp. 520–532, 2015.
- [28] G. Mannina, G. Di Bella, and G. Viviani, “Uncertainty assessment of a membrane bioreactor model using the GLUE methodology,” *Biochemical Engineering Journal*, vol. 52, no. 2–3, pp. 263–275, 2010.
- [29] J.-P. Vila and V. Wagner, “Predictive neuro-control of uncertain systems: design and use of a neuro-optimizer,” *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 39, no. 5, pp. 767–777, 2003.
- [30] S. Rómoli, M. E. Serrano, O. A. Ortiz, Vega J. R., and G. J. E. Scaglia, “Tracking control of concentration profiles in a fed-batch bioreactor using a linear algebra methodology,” *ISA Transactions*, vol. 57, no. 0, pp. 162–171, 2015.
- [31] S. Park and W. F. Ramirez, “Optimal production of secreted protein in fed-batch reactors,” *AIChE Journal*, vol. 34, no. 9, pp. 1550–1558, 1988.
- [32] X. Chen, W. Du, H. Tianfield, R. Qi, W. He, and F. Qian, “Dynamic optimization of industrial processes with nonuniform discretization-based control vector parameterization,” *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 4, pp. 1289–1299, 2014.
- [33] W. Gong, J. Liang, X. Kan, L. Wang, and A. M. Dobaie, “Robust state estimation for stochastic complex-valued neural networks with sampled-data,” *Neural Computing and Applications*, 2017.
- [34] S. Romoli, G. J. E. Scaglia, M. E. Serrano, S. A. Godoy, O. A. Ortiz, and J. R. Vega, “Control of a fed-batch fermenter based on a linear algebra strategy,” *IEEE Latin America Transactions*, vol. 12, no. 7, pp. 1206–1213, 2014.
- [35] J. M. Ali, N. H. Hoang, M. A. Hussain, and D. Dochain, “Review and classification of recent observers applied in chemical process systems,” *Computers and Chemical Engineering*, vol. 76, pp. 27–41, 2015.
- [36] H. Michalska and D. Q. Mayne, “Robust receding horizon control of constrained nonlinear systems,” *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 38, no. 11, pp. 1623–1633, 1993.
- [37] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, “Constrained model predictive control: stability and optimality,” *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 36, no. 6, pp. 789–814, 2000.
- [38] M. E. Serrano, S. A. Godoy, S. Rómoli, and G. J. E. Scaglia, “A numerical approximation-based controller for mobile robots with velocity limitation,” *Asian Journal of Control*, 2017.
- [39] S. G. Kulkarni, A. K. Chaudhary, S. Nandi, S. S. Tambe, and B. D. Kulkarni, “Modeling and monitoring of batch processes using principal component analysis (PCA) assisted generalized regression neural networks (GRNN),” *Biochemical Engineering Journal*, vol. 18, no. 3, pp. 193–210, 2004.
- [40] M. Von Stosch, R. Oliveira, J. Peres, and S. Feyo De Azevedo, “A novel identification method for hybrid (N)PLS dynamical systems with application to bioprocesses,” *Expert Systems with Applications*, vol. 38, no. 9, pp. 10862–10874, 2011.

- [41] Z. Xiong and J. Zhang, "Modelling and optimal control of fed-batch processes using a novel control affine feedforward neural network," *Neurocomputing*, vol. 61, no. 1-4, pp. 317–337, 2004.
- [42] F. G. Rossomando and C. M. Soria, "Identification and control of nonlinear dynamics of a mobile robot in discrete time using an adaptive technique based on neural PID," *Neural Computing and Applications*, vol. 26, no. 5, pp. 1179–1191, 2015.
- [43] R. F. Escobar, C. M. Astorga-Zaragoza, A. C. Tllez-Anguiano, D. Jurez-Romero, J. A. Hernandez, and G. V. Guerrero-Ramrez, "Sensor fault detection and isolation via high-gain observers: application to a double-pipe heat exchanger," *ISA Transactions*, vol. 50, no. 3, pp. 480–486, 2011.
- [44] Matlab™, *User Guide, 7.6.0.324*, The MathWorks, Inc, Torrance, CA, USA, 2008.
- [45] M. P. Aghababa, "Adaptive control of nonlinear complex Holling II predator-prey system with unknown parameters," *Complexity*, vol. 21, no. 6, pp. 260–266, 2016.
- [46] R. Tempo and H. Ishii, "Monte Carlo and Las Vegas randomized algorithms for systems and control," *European Journal of Control*, vol. 13, no. 2-3, pp. 189–203, 2007.
- [47] A. Noruzi, T. Banki, O. Abedinia, and N. Ghadimi, "A new method for probabilistic assessments in power systems, combining Monte Carlo and stochastic-algebraic methods," *Complexity*, vol. 21, no. 2, pp. 100–110, 2015.
- [48] H. Hao, K. Zhang, S. X. Ding, Z. Chen, and Y. Lei, "A data-driven multiplicative fault diagnosis approach for automation processes," *ISA Transactions*, vol. 53, no. 5, pp. 1436–1445, 2014.
- [49] C. D. Karlgaard and H. Shen, "Robust state estimation using desensitized divided difference filter," *ISA Transactions*, vol. 52, no. 5, pp. 629–637, 2013.
- [50] K. Michail, A. C. Zolotas, and R. M. Goodall, "Optimised sensor selection for control and fault tolerance of electromagnetic suspension systems: a robust loop shaping approach," *ISA Transactions*, vol. 53, no. 1, pp. 97–109, 2014.
- [51] M.-C. Pai, "Robust tracking and model following for uncertain time-delay systems with input nonlinearity," *Complexity*, vol. 21, no. 2, pp. 66–73, 2015.
- [52] Z. Chen and T. Zhang, "Long time behaviour of a stochastic model for continuous flow bioreactor," *Journal of Mathematical Chemistry*, vol. 51, no. 2, pp. 451–464, 2013.
- [53] C. Xu and S. Yuan, "An analogue of break-even concentration in a simple stochastic chemostat model," *Applied Mathematics Letters. An International Journal of Rapid Publication*, vol. 48, pp. 62–68, 2015.
- [54] MathWorks, 2017, <http://www.mathworks.com/help/simulink/slref/bandlimitedwhitenoise.html>.
- [55] V. Rajinikanth and K. Latha, "Tuning and retuning of PID controller for unstable systems using evolutionary algorithm," *ISRN Chemical Engineering*, vol. 2012, Article ID 693545, 11 pages, 2012.

Research Article

Dynamic Learning from Adaptive Neural Control of Uncertain Robots with Guaranteed Full-State Tracking Precision

Min Wang, Yanwen Zhang, and Huiping Ye

School of Automation Science and Engineering, Guangzhou Key Laboratory of Brain Computer Interaction and Applications, South China University of Technology, Guangzhou 510641, China

Correspondence should be addressed to Min Wang; auwangmin@scut.edu.cn

Received 21 March 2017; Accepted 30 April 2017; Published 14 August 2017

Academic Editor: Yanan Li

Copyright © 2017 Min Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A dynamic learning method is developed for an uncertain n -link robot with unknown system dynamics, achieving predefined performance attributes on the link angular position and velocity tracking errors. For a known nonsingular initial robotic condition, performance functions and unconstrained transformation errors are employed to prevent the violation of the full-state tracking error constraints. By combining two independent Lyapunov functions and radial basis function (RBF) neural network (NN) approximator, a novel and simple adaptive neural control scheme is proposed for the dynamics of the unconstrained transformation errors, which guarantees uniformly ultimate boundedness of all the signals in the closed-loop system. In the steady-state control process, RBF NNs are verified to satisfy the partial persistent excitation (PE) condition. Subsequently, an appropriate state transformation is adopted to achieve the accurate convergence of neural weight estimates. The corresponding experienced knowledge on unknown robotic dynamics is stored in NNs with constant neural weight values. Using the stored knowledge, a static neural learning controller is developed to improve the full-state tracking performance. A comparative simulation study on a 2-link robot illustrates the effectiveness of the proposed scheme.

1. Introduction

In the past decades, the force/position tracking control problem of robots has attracted wide attention in both theory and applications [1, 2]. In the early stage of robotic tracking control, the system model is usually assumed to be accurately known, and the corresponding model-based control method has been proposed in [3, 4]. Along with the diversity of robot working environment and the complexity of the robot's structure, the force/position tracking control has been studied in different kinds of uncertainties. Nowadays, ignoring uncertainties to simplify control design may cause the large steady-state errors or/and poor transient response [5]. For the case of parametric uncertainties, adaptive control method has been presented in [6, 7] to make robots adapt the changing control environment. In order to enhance system robustness on the uncertain parameters in the presence of external disturbances, sliding mode control [8, 9] has been proposed to obtain the desired robotic tracking control performance. Owing to the universal approximation property [10–17],

a great number of intelligent control schemes, such as adaptive neural/fuzzy control, have been developed for controlling robotic systems with uncertain nonlinearities [18–22].

Although intelligent control of robotic systems has attracted considerable attention in the past few years, relatively few robot control methods could achieve human-like performance in a dynamic and uncertain environment. It is well known that intelligent control was initially inspired by the learning and control abilities of human beings, thus intelligent control should at least possess the aforementioned properties “learning by doing” and “doing with the learned knowledge” [23–26]. But most existing intelligent control schemes can only ensure the stability of closed-loop systems without being able to achieve the information acquisition and storage and utilization of unknown system dynamics. This means that the existing intelligent control schemes do not solve the accurate convergence of estimated parameters, which usually needs to guarantee the exponential stability of the derived closed-loop system. However, it is extremely difficult for uncertain nonlinear systems to verify

the exponential stability of the derived closed-loop system. Recently, a deterministic learning method was proposed in [27] using RBF NNs for a second-order Brunovsky system, where the derived closed-loop system was described by a class of linear time-varying systems. By verifying that RBF NNs satisfied persistent excitation (PE) condition, the convergence of partial neural weights and accurate neural approximation of unknown system dynamics were guaranteed in [27] because of the exponential convergence of the derived linear time-varying closed-loop systems. The deterministic learning method was further extended to n th-order Brunovsky systems with an unknown affine/nonaffine term [28, 29], where the derivative of unknown affine terms was assumed to be bounded. By combining backstepping with a system decomposition strategy, an elegant dynamic learning method was proposed to cope with the learning and control problem of third-order strict-feedback systems [30]. By combining dynamic surface control technology [31], the result in [30] was further extended to n th-order strict-feedback systems. The deterministic learning method was also applied in many physical systems such as marine surface vessels [32, 33] and robot manipulators [34].

Recently, the deterministic learning methods are mainly used to solve the learning and control problem for single input single output nonlinear systems without any constraint. In practice, there are many different kinds of constraints in most of physical systems, such as output or state constraints, tracking performance constraints. The violation of the constraints may cause severe performance degradation, safety problem, or system damage [35]. Therefore, it is of great importance for solving the control and learning problem of constrained systems. Based on Lyapunov theorem, a barrier Lyapunov function (BLF) method has been presented to solve output constraints for strict-feedback nonlinear systems [36], output feedback nonlinear systems [37], flexible systems [38], and robotic manipulator [35, 39]. The BLF-based methods were also extended to solve state constraints [40–42]. Although the aforementioned results on the output or state constraints can guarantee that system outputs or states converge to a predefined bounded set, the predefined performance requirements on the convergence rate, maximum overshoot, and steady-state error have not been studied fully. The predefined performance issue is an extremely challenging problem. Recently, adaptive neural prescribed performance controller was proposed in [43, 44] for feedback linearizable nonlinear systems by means of transformation functions. The proposed method was also developed to deal with the constrained output tracking control problem in many applications such as robotic systems [45], nonlinear servo mechanisms [46], marine surface vessels [33], nonlinear stochastic large-scale systems with actuator faults [47], and switched nonlinear systems [48]. To solve partial tracking error constraints, a fuzzy dynamic surface control design was developed in [49, 50] for a class of strict-feedback nonlinear systems by transforming the state tracking errors into new virtual variables. However, the existing control schemes, such as [35–51], can only guarantee the stability of closed-loop systems with different constraints, which are not capable of achieving the learning of unknown system dynamics. The

main reason is that the derived closed-loop error system is extremely complex, such that its exponential convergence is difficult to be verified using the existing stability analysis tools. To solve this problem, a neural learning control with the output tracking error constraint was presented in [52, 53] for a class of nonlinear systems. These methods proposed in [52, 53] cannot be adopted to deal with the dynamic learning problem for multi-input and multioutput nonlinear systems with full-state tracking error constraints.

Based on the above discussions, this paper proposes a novel dynamic learning method for a multi-input and multi-output n -link robot with full-state tracking error constraints and a mild assumption. To prevent the violation of the full-state tracking error constraints, performance functions are firstly introduced to characterize the transient and steady-state performance of full-state tracking errors. And then, using a nonlinear transformation method, the constrained tracking control problem is effectively transformed into the stabilization problem of equivalent unconstrained transformation error systems. By combining backstepping and Lyapunov stability, a novel adaptive neural control scheme is proposed to guarantee the uniformly ultimate boundedness of all closed-loop signals and the prescribed full-state tracking error performance. It should be pointed out that the proposed control scheme is different from the traditional backstepping design. In our control design, the correlative interconnection term can not be compensated in next step of backstepping design because of the full-state tracking error constraints. To overcome the difficulty, two independent Lyapunov functions are constructed in each step of backstepping design. Based on the independent Lyapunov functions, the unconstrained transformation error of the link angular velocity tracking error is firstly proved to be bounded, and the corresponding link angular velocity tracking error is further proved to satisfy the prescribed performance. Invoking the boundedness of the link angular velocity tracking error, we backward derive the link angular position tracking error to satisfy the prescribed performance. By means of the tracking convergence in the steady-state control process, the regression subvector consisting of the RBFs along the recurrent tracking orbit satisfies the partial PE condition. Subsequently, an appropriate state transformation is introduced to transform the closed-loop system into a linear time-varying (LTV) system with small perturbed terms. With the perturbation theory of LTV system, the knowledge of the closed-loop robotic dynamics can be accurately stored by RBF NNs with constant weight values. The stored knowledge can be reused to develop a static neural learning controller for achieving the better control performance with smaller transient-state tracking errors, smaller control gains, and less computational time.

The rest of this paper is organized as follows. Section 2 introduces the system formulation, the full-state tracking error transformation, and some useful preliminaries about RBF networks. In Section 3, a novel adaptive neural control design is proposed for rigid robotic manipulators with constrained full-state tracking performances. Neural learning control is developed in Section 4, which achieves the knowledge acquisition, storage, and utilization of unknown robotic dynamics. In Section 5, simulation studies on 2-link robotic

system are given to show the effectiveness of the proposed method. Section 6 includes the conclusions of the paper.

2. Problem Formulation and Preliminaries

2.1. Uncertain Robotic System. The dynamics of an n -link robotic system is described in the following Lagrange form [1]:

$$M(q)\ddot{q} + V_m(q, \dot{q})\dot{q} + G(q) + F(\dot{q}) + d(t) = \tau, \quad (1)$$

where $q, \dot{q}, \ddot{q} \in R^n$ are the angular position, velocity, and acceleration, respectively, $\tau \in R^n$ refers to the input torque, $M(q) \in R^{n \times n}$ is an unknown symmetric positive definite inertia matrix, $V_m(q, \dot{q}) \in R^{n \times n}$ denotes unknown centripetal and Coriolis torques, $G(q) \in R^n$ is an unknown gravitational force vector, $F(\dot{q}) \in R^n$ is an unknown friction vector, and $d(t) \in R^n$ denotes unknown disturbances,

Property 1 (see [54]). The matrix $\dot{M}(q) - 2V_m(q, \dot{q})$ is skew-symmetric.

Assumption 1. The unknown external disturbance $d(t)$ is bounded, that is, there exists a constant $d^* \in R^+$ such that $\|d(t)\| < d^*$.

In this paper, we choose $y_d = x_{d1} \in R^n$ as a recurrent reference trajectory of the angular position q , which is generated by the following reference model:

$$\begin{aligned} \dot{x}_{d1} &= x_{d2}, \\ \dot{x}_{d2} &= F_d(x_d), \end{aligned} \quad (2)$$

where $x_d = [x_{d1}^T, x_{d2}^T]^T \in R^{2n}$ is the state vector of the reference model, which is assumed to be recurrent signals, and $F_d(x_d)$ is a known smooth function. The reference orbit along the given initial condition $x_d(0)$ is defined as $\varphi_d(x_d(0))$. In this paper, $\varphi_d(x_d(0))$ is assumed to be a recurrent motion.

2.2. Full-State Tracking Constraints. Define full-state tracking errors as $e_1 = q - y_d = [e_{11}, e_{12}, \dots, e_{1n}]^T \in R^n$ and $e_2 = \dot{q} - \dot{y}_d = [e_{21}, e_{22}, \dots, e_{2n}]^T \in R^n$, where $\alpha_1 \in R^n$ are continuously differentiable virtual control signals. For guaranteeing the prescribed transient and steady-state bounds of full-state tracking errors, e_1 and e_2 need to satisfy the following predefined condition:

$$-\underline{\delta}_{ij}\rho_i(t) < e_{ij} < \bar{\delta}_{ij}\rho_i(t), \quad (3)$$

where $i = 1, 2, j = 1, 2, \dots, n$, and $\underline{\delta}_{ij}, \bar{\delta}_{ij}$ are positive design constants and $\rho_i(t)$ is a smooth, strictly positive, bounded, and decreasing function, which satisfies $\lim_{t \rightarrow \infty} \rho_i(t) > 0$ and is called performance function. In this paper, $\rho_i(t)$ is chosen as the following exponential performance function:

$$\rho_i(t) = (\rho_{i0} - \rho_{i\infty})e^{-\kappa_i t} + \rho_{i\infty}, \quad \forall t \geq 0, \quad (4)$$

where $\rho_{i0}, \rho_{i\infty}$ and κ_i are positive design constants.

For any given initial condition $e_i(0)$, these design constants $\rho_{i0}, \underline{\delta}_{ij}$ and $\bar{\delta}_{ij}$ can be chosen appropriately such that $e_i(0)$ satisfies the predefined condition (3). From (3) and (4), the different selection of these design parameters, $\rho_{i0}, \rho_{i\infty}, \kappa_i, \underline{\delta}_{ij}$ and $\bar{\delta}_{ij}$, can obtain different performance requirements on the tracking error e_{ij} .

2.3. RBF Neural Networks. The RBF NNs can be described by $f_{nn}(Z) = W^T S(Z)$, where $Z \in \Omega_Z \subset R^q$ are NN input variables with $q > 1$ being NN input dimension and Ω_Z being a compact set, $W = [w_1, \dots, w_N]^T \in R^N$ is the weight vector, $N > 1$ is the NN node number, and $S(Z) = [s_1(\|Z - \mu_1\|), \dots, s_N(\|Z - \mu_N\|)]^T$ is the radial basis function vector. The RBF neural networks have the following desired properties.

(1) Universal Approximation

Lemma 2 (see [55]). Using sufficiently large node number N , the RBF NNs $W^T S(Z)$ can approximate any smooth function $f(Z) : \Omega_Z \rightarrow R^q$ over a compact set Ω_Z to any arbitrary accuracy as

$$f(Z) = W^{*T} S(Z) + \eta(Z), \quad (5)$$

where W^* is the ideal weight vector of W and $\eta(Z)$ is any small approximation error which satisfies $|\eta(Z)| \leq \eta^*$.

(2) Spatially Localized Approximation Property. It should be noted that the radial basic function satisfies $s_i(\|Z - \mu_i\|) \rightarrow 0$ when $\|Z - \mu_i\| \rightarrow \infty$. This property shows that the network output is only locally affected by each basis function. Therefore, any smooth function $f(Z)$ over a compact set Ω_Z can be approximated using a limited number of neurons, which are located in a local region along bounded input trajectory Z :

$$f(Z) = W_{\zeta}^{*T} S_{\zeta}(Z) + \eta_{\zeta}(Z), \quad (6)$$

where $S_{\zeta}(Z) \in R^{N_{\zeta}}$, $N_{\zeta} < N$ is the subvector of $S(Z)$, composed of RBFs that are close to the trajectory Z , $W_{\zeta}^* \in R^{N_{\zeta}}$ is the corresponding subvector of W^* , and approximation errors $\eta_{\zeta}(Z) = f(Z) - W_{\zeta}^{*T} S_{\zeta}(Z)$, $\bar{\zeta}$ are defined as the region far away from the trajectory Z , which means $\|S_{\bar{\zeta}}(Z)\| \rightarrow 0$. Therefore, $\eta_{\zeta}(Z)$ is close to $\eta(Z)$, which is any small value.

(3) Partial PE Condition. Persistent excitation plays a key role in accurate convergence of neural weight estimator. To clearly show that RBF NNs satisfy the PE property, a definition of PE condition is given as follows:

Definition 3 (see [55]). A continuous, uniformly bounded, vector-valued function $S : [0, \infty] \rightarrow R^m$ is said to satisfy the persistent excitation condition, if there exist positive constants α_1, α_2 , and δ such that

$$\alpha_1 \|c\|^2 \leq \int_{t_0}^{t_0 + \sigma} |S(\tau)^T c|^2 d\mu(\tau) \leq \alpha_2 \|c\|^2, \quad \forall t_0 \geq 0, \quad (7)$$

holds for every constant vector $c \in R^N$, where μ is a positive, Σ -finite Borel measure on $[0, \infty]$.

Lemma 4 (see [27]). *Consider any recurrent orbit $Z \in R^q$, remaining in a compact set Ω_Z with $\Omega_Z \subset R^q$. Then, for the RBF network $W^T S(Z)$ with centers placed on a regular lattice, which is large enough to cover the compact set Ω_Z , the regressor subvector $S_r(Z)$ in (6) (rather than the entire regressor vector $S(Z)$) is persistently exciting.*

3. Adaptive Neural Control with Full-State Tracking Error Constraints

In this section, a novel stable adaptive neural tracking control scheme will be developed for the system (1) with full-state tracking error constraints (3) using nonlinear error transformations, independent Lyapunov functions, and backstepping. The proposed control scheme will guarantee that all the signals in the closed-loop system are ultimately bounded and the full-state tracking errors satisfy the prescribed performances (3).

In order to transform the constrained tracking control problem (3) into an equivalent unconstrained one, a performance transformation method is introduced as follows:

$$e_{ij} = \rho_i(t) T_{ij}(c_{ij}), \quad (8)$$

where $T_{ij}(c_{ij})$ is a smooth and strictly increasing function and satisfies $T_{ij}(\cdot) : (-\infty, \infty) \rightarrow (-\bar{\delta}_{ij}, \bar{\delta}_{ij})$. In this paper, the transformation function $T_{ij}(c_{ij})$ is constructed as follows:

$$T_{ij}(c_{ij}) = \frac{\bar{\delta}_{ij} e^{c_{ij}} - \underline{\delta}_{ij} e^{-c_{ij}}}{e^{c_{ij}} + e^{-c_{ij}}}. \quad (9)$$

Owing to the properties $T_{ij}(c_{ij})$ and $\rho_i(t) \geq \rho_{i\infty} > 0$, the inverse transformation of $T_{ij}(c_{ij})$ exists and is well defined as follows:

$$c_{ij} = T_{ij}^{-1} \left[\frac{e_{ij}}{\rho_i(t)} \right]. \quad (10)$$

From (3) and (10), if c_{ij} is verified to be bounded, then we can obtain that e_{ij} satisfies the predefined error performance (3). Differentiating c_{ij} yields

$$\dot{c}_{ij} = r_{ij} \left[\dot{e}_{ij} - \frac{\dot{\rho}_i(t)}{\rho_i(t)} e_{ij} \right], \quad (11)$$

where $r_{ij} = \partial T_{ij}^{-1}[e_{ij}/\rho_i(t)] / [\rho_i(t) \partial(e_{ij}/\rho_i(t))]$. Next, define the transformed unconstrained error vector $\xi_i = [c_{i1}, c_{i2}, \dots, c_{in}]^T \in R^n$, $i = 1, 2$. It follows from (11) that

$$\dot{\xi}_i = \Upsilon_i (\dot{e}_i - \Psi_i e_i), \quad (12)$$

where $\Psi_i = \dot{\rho}_i(t)/\rho_i(t)$ and $\Upsilon_i = \text{diag}\{r_{i1}, r_{i2}, \dots, r_{in}\} \in R^{n \times n}$. Notice that $e_1 = q - y_d$ and $e_2 = \dot{q} - \alpha_1$, using system (1), and thus the dynamics of the transformed error vector can be rewritten as

$$\dot{\xi}_1 = \Upsilon_1 (e_2 + \alpha_1 - x_{d2} - \Psi_1 e_1), \quad (13)$$

$$\dot{\xi}_2 = \Upsilon_2 M^{-1}(q) [F(q, \dot{q}, \vartheta) + \tau - d(t)], \quad (14)$$

where $F(q, \dot{q}, \vartheta) = -V_m(q, \dot{q})\dot{q} - G(q) - F(\dot{q}) - M(q)\vartheta$ is an unknown smooth function vector and $\vartheta = \dot{\alpha}_1 + \Psi_2 e_2$ is a computable variable. Next, a control scheme will be developed for the system (13)-(14) based on backstepping.

Step 1. For the transformed error subsystem (13), design a virtual control

$$\alpha_1 = -k_1 \xi_1 + x_{d2} + \Psi_1 e_1. \quad (15)$$

Then, we have

$$\dot{\xi}_1 = \Upsilon_1 (e_2 - k_1 \xi_1), \quad (16)$$

where $k_1 = k_{10} + k_{11}$ with $k_{10} > 0$ and $k_{11} > 0$.

Constructing the following Lyapunov function candidate:

$$V_1 = \frac{\xi_1^T \xi_1}{2} \quad (17)$$

whose derivative along (16) yields

$$\dot{V}_1 = -k_1 \xi_1^T \Upsilon_1 \xi_1 + \xi_1^T \Upsilon_1 e_2. \quad (18)$$

Remark 5. From (18), the boundedness of the transformation error ξ_1 depends on the boundedness of the link angular velocity tracking error e_2 . In the traditional backstepping design, the term $\xi_1^T \Upsilon_1 e_2$ is called the correlative interconnection term, which can be usually compensated in the next step of backstepping. However, in our control design, it is impossible to deal with $\xi_1^T \Upsilon_1 e_2$ in Step 2. The main reason is that the full-state tracking error constraints are considered in this paper, which derives the transformation error ξ_2 in Step 2, instead of the traditional error e_2 . Therefore, it is difficult to construct the Lyapunov V_2 with ξ_2 to compensate for $\xi_1^T \Upsilon_1 e_2$; see Step 2 for details.

Step 2. By adding and subtracting $\Upsilon_2^{-1} V_m(q, \dot{q}) \xi_2$, the transformed error subsystem (14) can be rewritten as

$$\dot{\xi}_2 = \Upsilon_2 M^{-1} [\Phi(Z) + \tau - d(t) - \Upsilon_2^{-1} V_m(q, \dot{q}) \xi_2], \quad (19)$$

where $Z = [q^T, \dot{q}^T, \xi_2^T, \vartheta^T]^T$, and

$$\Phi(Z) = F(q, \dot{q}, \vartheta) + \Upsilon_2^{-1} V_m(q, \dot{q}) \xi_2. \quad (20)$$

It should be pointed out that the term $\Upsilon_2^{-1} V_m(q, \dot{q}) \xi_2$ introduced in (19) facilitates the stability analysis based on the property (1). Since $F(q, \dot{q})$, M , and $V_m(q, \dot{q})$ are unknown smooth function vectors, $\Phi(Z)$ is also unknown and smooth, which can not directly be used to construct the controller. To solve this problem, the unknown dynamics $\Phi(Z)$ are approximated by RBF NN (5) in this paper. Then, we have

$$\dot{\xi}_2 = \Upsilon_2 M^{-1} [W^{*T} S(Z) + \eta(Z) + \tau - \Upsilon_2^{-1} V_m(q, \dot{q}) \xi_2 - d(t)], \quad (21)$$

where the approximation error $\eta(Z) \in R^n$ satisfies $\|\eta(Z)\| \leq \eta^*$ and $W^* = [W_1^*, \dots, W_n^*] \in R^{N \times n}$ is the unknown optimal NN weight vector with NN node number $N > 1$, and $S(Z) = [s_1(Z), s_2(Z), \dots, s_N(Z)]^T \in R^N$ is a radial basis function vector. Define \widehat{W} as the estimated weight values of W^* , and let $\widetilde{W} = \widehat{W} - W^*$ be the corresponding estimated error. Then, design the adaptive neural control law as

$$\tau = -k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 - \widehat{W}^T S(Z) \quad (22)$$

and construct the neural weight updated law as

$$\dot{\widehat{W}} = \dot{\widetilde{W}} = \Gamma S(Z) (Y_2 \xi_2)^T - \sigma \Gamma \widehat{W}, \quad (23)$$

where k_{20} and k_{21} are positive design constants, $\Gamma > 0$ is positive diagonal matrix, and $\sigma > 0$ is a small value, which is used to improve the robustness of the adaptive controller (22).

Substituting (22) into (21), we have

$$\begin{aligned} \dot{\xi}_2 &= Y_2 M^{-1} [-k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 - \widehat{W}^T S(Z) \\ &\quad - Y_2^{-1}V_m(q, \dot{q})\xi_2 + \bar{\eta}], \end{aligned} \quad (24)$$

where $\bar{\eta} = \eta(Z) - d(t)$. Noting that the inertia matrix M is positive definite, thus we construct the following Lyapunov function candidate:

$$V_2 = \frac{1}{2}\xi_2^T M \xi_2 + \frac{1}{2}\text{tr}[\widehat{W}^T \Gamma^{-1} \widehat{W}]. \quad (25)$$

In what follows, one of our main results will be shown by the following theorem.

Theorem 6 (stability and tracking). *Consider the closed-loop system consisting of the robotic system (1), the bounded reference trajectory (2), the full-state tracking performance condition (3), the transformed error (10), the proposed adaptive neural control law (22) with the virtual control law (15), and the weight updated law (23). Assume the given bounded initial conditions satisfy the condition (3) (this condition can be satisfied by choosing proper designed parameters $\underline{\delta}_{ij}$, $\bar{\delta}_{ij}$, and ρ_{i0}). Then, we have that*

- (1) all signals of the closed-loop system remain uniformly ultimately bounded
- (2) the constrained full-state tracking errors e_{ij} satisfy the prescribed performance (3) and they converge to a small residual set of zero in a finite time T .

Proof. Differentiating V_2 in (25) with respect to time and using (18), (24) yields

$$\begin{aligned} \dot{V}_2 &= \xi_2^T Y_2 [-k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 - \widehat{W}^T S(Z) + \bar{\eta}] \\ &\quad + \frac{1}{2}\xi_2^T (\dot{M} - 2V_m(q, \dot{q}))\xi_2 + \text{tr}[\widehat{W}^T \Gamma^{-1} \dot{\widehat{W}}]. \end{aligned} \quad (26)$$

Using the Property 1 and substituting (23) into (26), the derivation of V_2 is rewritten as

$$\begin{aligned} \dot{V}_2 &= \text{tr}[-\sigma \widehat{W}^T \widehat{W}] \\ &\quad + \xi_2^T Y_2 [-k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 + \bar{\eta}]. \end{aligned} \quad (27)$$

Subsequently, using the appropriate inequality and combining Assumption 1, we have

$$\xi_2^T Y_2 \bar{\eta} \leq k_{21} \xi_2^T Y_2^2 \xi_2 + \frac{\bar{\eta}^{*2}}{4k_{21}}, \quad (28)$$

$$\text{tr}[-\sigma \widehat{W}^T \widehat{W}] \leq -\frac{\sigma}{2} \|\widehat{W}\|_F^2 + \frac{\sigma}{2} \|W^*\|_F^2,$$

where $\bar{\eta}^{*2} = \eta^* + d^*$ is a positive constant. Substituting (28) into (27) gives

$$\begin{aligned} \dot{V}_2 &\leq -k_{20} \xi_2^T \xi_2 - \frac{\sigma}{2} \|\widehat{W}\|_F^2 + \frac{\sigma}{2} \|W^*\|_F^2 + \frac{\bar{\eta}^{*2}}{4k_{21}} \\ &\leq -a_2 V + b_2, \end{aligned} \quad (29)$$

where $a_2 = \min\{k_{20}/\lambda_{\max}(M), \sigma\lambda_{\min}(\Gamma)\}$, $b_2 = (\sigma/2)\|W^*\|_F^2 + \bar{\eta}^{*2}/4k_{21}$. Let $\beta_2 = b_2/a_2$; we have

$$V_2 \leq (V_2(0) - \beta_2) \exp(-a_2 t) + \beta_2. \quad (30)$$

□

From the definition (25) of V_2 , the transformed error ξ_2 and the neural weight error \widehat{W} are bounded. Noticing that $\xi_2 = [\varsigma_{21}, \varsigma_{22}, \dots, \varsigma_{2n}]^T \in R^n$, and the error transformed relationship (8)–(10), thus we obtain that the tracking error e_2 is bounded and satisfies the prescribed tracking error constraints (3). This means $\|e_2\| \leq \delta^*$, δ^* is any small value depending on design parameters $\underline{\delta}_{ij}$, $\bar{\delta}_{ij}$, ρ_{20} , and $\rho_{2\infty}$. Using the bounded property of e_2 , we can obtain that

$$\xi_1^T Y_1 e_2 \leq k_{11} \xi_1^T Y_1 \xi_1 + \frac{\max(r_{i1}) \delta^{*2}}{4k_{11}}. \quad (31)$$

Noting $k_1 = k_{10} + k_{11}$, thus (18) can be rewritten as

$$\dot{V}_1 \leq -k_{10} \xi_1^T Y_1 \xi_1 + \frac{\max(r_{i1}) \delta^{*2}}{4k_{11}} \leq -a_1 V_1 + b_1, \quad (32)$$

where $a_1 = \min\{2k_{10}/\lambda_{\max}(Y)\}$, $b_1 = \max(r_{i1})\delta^{*2}/4k_{11}$. Further, let $\beta_1 = b_1/a_1$; we have

$$V_1 \leq (V_1(0) - \beta_1) \exp(-a_1 t) + \beta_1. \quad (33)$$

Since $V_1 = \xi_1^T \xi_1/2$, it follows from (33) that ξ_1 is bounded. Similarly, we can obtain that the tracking error e_1 is bounded and satisfies the prescribed tracking error constraints (3). By combining the boundedness of x_{d1} , x_{d2} and the deigned bounded function Y , Ψ_1 , it can be verified that q and α_1 are bounded. Because of $e_2 = \dot{q} - \alpha_1$, \dot{q} is also bounded. Because the desired weight W^* is bounded, the weight estimate \widehat{W} is also bounded. From (22), we can verify that τ is also bounded. Hence, all closed-loop signals are uniformly ultimately bounded.

Moreover, from (17), (25), (30), and (33), we can obtain the convergence region of the transformed error ξ_1, ξ_2 as follows:

$$\begin{aligned} \sum_{i=1}^n \varsigma_{1i}^2 &\leq 2\beta_1 + 2V_1(0) \exp(-a_1 t), \\ \sum_{i=1}^n \varsigma_{2i}^2 &\leq \frac{2\beta_2}{\lambda_{\min}(M)} + \frac{2V_2(0) \exp(-a_2 t)}{\lambda_{\min}(M)}. \end{aligned} \quad (34)$$

By choosing $\varphi_1 > \sqrt{2\beta_1}$ and $\varphi_2 > \sqrt{2\beta_2/\lambda_{\min}(M)}$, there exists a finite time T , such that for $\forall t > T$ the transformed error ς_{ij} satisfies

$$|\varsigma_{ij}| < \varphi_i, \quad i = 1, 2, \quad j = 1, 2, \dots, n. \quad (35)$$

Subsequently, the convergence region after a finite time T can be constructed as follows:

$$\Omega_\varsigma = \{\varsigma_{ij} \mid |\varsigma_{ij}| < \varphi_i, X_d \in \Omega_d\}, \quad (36)$$

where φ can be adjusted to be arbitrarily small if we choose appropriate controller parameters k_{i0}, k_{i1} , and σ .

From (36), the transformed error ς_{ij} converges to a small residual set Ω_ς in a finite time T . Owing to the convergence of ς_{ij} and the error transformed relationship (8)–(10), the full-state constrained tracking errors $e_{ij}(t)$ satisfy (3) in a finite time T . From (3) and (4), the tracking error e_i exponentially converges to the interval $[-\underline{\delta}_{ij}\rho_{i\infty}, \delta_{ij}\rho_{i\infty}]$, which can be adjusted to be a small residual set of zero by choosing the appropriate design parameters $\rho_{i\infty}, \underline{\delta}_{ij}, \bar{\delta}_{ij}$.

Remark 7. In order to verify the boundedness of transformation errors ξ_1 and ξ_2 , two independent Lyapunov functions V_1 and V_2 are constructed in Steps 1 and 2. Using the appropriate inequality technology, we firstly prove the transformation error ξ_2 is bounded. Subsequently, the boundedness of the tracking error e_2 is indirectly verified based on the error transformed relationship (8)–(10). As indicated by the Remark 5, the boundedness of e_2 backward derives the boundedness of the transformation error ξ_1 . Similarly, e_1 can be verified to satisfy the predefined tracking performance (3).

Remark 8. In order to verify the boundedness of transformation errors ξ_1 and ξ_2 , two independent Lyapunov functions V_1 and V_2 are constructed in Steps 1 and 2. Using the appropriate inequality technology, we firstly prove the transformation error ξ_2 is bounded. Although the boundedness of ξ_2 depends on W^* in (30) which may be large, the tracking error e_2 can still satisfy the prescribed performance (3) based on the error transformed relationship (8)–(10). Subsequently, the boundedness of e_2 backward derives the boundedness of the transformation error ξ_1 . Similarly, e_1 can be verified to satisfy the predefined tracking performance (3).

4. Neural Learning Control

Based on the stable adaptive neural control scheme developed in Section 3, this section will use the spatially localized

approximation ability of RBF NNs to achieve the knowledge acquisition and storage of the unknown system dynamics $\Phi(Z)$. And then, the stored knowledge will be reused to develop a neural learning controller so that the improved control performance of the robotic system (1) can be achieved for the same or a similar control task.

4.1. Knowledge Acquisition, Expression, and Storage. In this section, the regression subvector $S_\zeta(Z)$ of RBF NN is firstly verified to satisfy the PE condition, which is key to achieve the exponential convergence of NN weight values \widehat{W} and the accurate NN approximation of the unknown system dynamics $\Phi(Z)$. From Lemma 4, the NN input vector $Z = [q^T, \dot{q}^T, \xi_2^T, \vartheta^T]^T$ needs to be verified as recurrent signals, so that the regression subvector $S_\zeta(Z)$ along the input orbit Z satisfies partial PE condition. Based on Theorem 6, it can be obtained that the system output q converges to a small neighborhood of y_d for $\forall t > T$. Since y_d is a recurrent signal, q is also recurrent. Since $\dot{q} = \alpha_1 + e_2$, $\alpha_1 = -k_1\xi_1 + x_{d2} + \Psi_1 e_1$, e_1 , and ξ_1 are very small values, \dot{q} is recurrent with the same period as x_{d2} . In the steady-state control, ξ_2 is small and recurrent. Noting that $\vartheta = \dot{\alpha}_1 + \Psi_2 e_2$, which is a function of the variables e_i, ξ_i, x_i , and x_{di} , so it can be recurrently verified as a recurrent signal. According to Lemma 4, the regression subvector $S_\zeta(Z)$ satisfies partial PE condition.

Using the spatially localized approximation ability of RBF NNs, the closed-loop system from (23) and (24) can be given by

$$\begin{aligned} \dot{\xi}_2 &= G(Y_2, q) \left[-K_2(t) \xi_2 - \widehat{W}_\zeta^T S_\zeta(Z) + \bar{\eta}_\zeta \right], \\ \dot{\widehat{W}}_\zeta &= \Gamma_\zeta S_\zeta(Z) (Y_2 \xi_2)^T - \sigma \Gamma_\zeta \widehat{W}_\zeta, \\ \dot{\widehat{W}}_{\bar{\zeta}} &= \Gamma_{\bar{\zeta}} S_{\bar{\zeta}}(Z) (Y_2 \xi_2)^T - \sigma \Gamma_{\bar{\zeta}} \widehat{W}_{\bar{\zeta}}, \end{aligned} \quad (37)$$

where $K_2(t) = k_{20}Y_2^{-1} + k_{21}Y_2 + Y_2^{-1}V_m(q, \dot{q})$, $G(Y_2, q) = Y_2 M^{-1}$, which is a positive definite and symmetric matrix, $S_\zeta(Z)$ is the subvector of $S(Z)$, which is composed of RBFs that are close to the reference orbit Z , $\widehat{W}_\zeta = [\widehat{W}_{\zeta_1}, \widehat{W}_{\zeta_2}, \dots, \widehat{W}_{\zeta_n}] \in R^{\zeta \times n}$ is the corresponding estimated weight subvector with $\widehat{W}_{\zeta_i} \in R^\zeta$ and $0 < \zeta < N$, $\bar{\zeta}$ denotes the region far away from the orbit Z , and $\bar{\eta}_\zeta = \bar{\eta} - \widehat{W}_{\bar{\zeta}}^T S_{\bar{\zeta}}(Z)$ are the approximate errors along the reference orbit.

Theorem 9. Consider the closed-loop system consisting of the robotic system (1) with Assumption 1, the bounded reference model (2), full-state tracking error constrained condition (3), the transformation error (10) and adaptive neural control law (22) with the virtual control law (15), and the NN updated law (23). Assume the given bounded initial conditions satisfy (3) and $\widehat{W}_i(0) = 0, 1 \leq i \leq n$. Then, for any recurrent orbit $\phi_d(x_d(t))|t \geq 0$, we have that the NN weight estimates \widehat{W}_i exponentially converge to a small neighborhood of optimal values W_i^* after $t \geq T$, and the corresponding system dynamics

$\Phi_i(Z)$ along recurrent signals Z is approximated by the stored neural knowledge $\overline{W}_i^T S(Z)$ as:

$$\Phi_i(Z) = \overline{W}_i^T S(Z) + \eta_{i\overline{W}}(Z), \quad (38)$$

where $\eta_{i\overline{W}}(Z)$ approaches the desired error η_i^* , and the constant weight values \overline{W}_i are chosen as

$$\overline{W}_i = \text{mean}_{t \in [t_{ai}, t_{bi}]} \widehat{W}_i(t) = \frac{1}{t_{bi} - t_{ai}} \int_{t_{ai}}^{t_{bi}} \widehat{W}_i(s) ds \quad (39)$$

with $[t_{ai}, t_{bi}]$, $1 \leq i \leq n$, $t_{bi} > t_{ai} > T$ representing a time segment in a steady-state stage.

Proof. Up to now, we have verified that $S_\zeta(Z)$ satisfies the PE condition. Furthermore, in order to achieve the exponential convergence of neural weight estimates \widehat{W}_i , the closed-loop system (37) needs to be transformed into a class of linear time-varying (LTV) systems with small perturbations based on Lemma 4.6 given in [56]. From (37), the perturbation term $G(Y_2, q)\overline{\eta}_\zeta$ may be very large. The main reasons lie in the fact that the term $\overline{\eta}_\zeta = \overline{\eta} - \overline{W}_\zeta^T S_\zeta(Z) = \eta(Z) - d(t) - \overline{W}_\zeta^T S_\zeta(Z)$ may be large with the possible large $d(t)$ and the term $G(Y_2, q) = Y_2 M^{-1}(q)$ may be also large due to the possible large Y_2 and $M^{-1}(q)$. Noting $G(Y_2, q)$ is a positive definite and symmetric matrix, a state transformation $E_2 = G^{-1}(Y_2, q)\xi_2/M_s$ is introduced to obtain the following class of LTV systems:

$$\begin{bmatrix} \dot{E}_2 \\ \dot{\widehat{W}}_{\zeta 1} \\ \vdots \\ \dot{\widehat{W}}_{\zeta n} \end{bmatrix} = \begin{bmatrix} A(t) & B^T(t) \\ -C(t) & 0 \end{bmatrix} \begin{bmatrix} E_2 \\ \widehat{W}_{\zeta 1} \\ \vdots \\ \widehat{W}_{\zeta n} \end{bmatrix} + \begin{bmatrix} \overline{\eta}_\zeta \\ M_s \\ -\sigma \Gamma_\zeta \widehat{W}_{\zeta 1} \\ \vdots \\ -\sigma \Gamma_\zeta \widehat{W}_{\zeta n} \end{bmatrix}, \quad (40)$$

where G is abbreviation of $G(Y_2, q)$, and

$$\begin{aligned} A(t) &= -K_2(t)G + \dot{G}^{-1}G \in R^{n \times n}, \\ B^T(t) &= -\frac{1}{M_s} \text{diag}\{S_\zeta^T, \dots, S_\zeta^T\} \in R^{n \times \zeta}, \\ C(t) &= -M_s \overline{\Gamma}_\zeta \overline{S}_\zeta(Z) Y_2 G \in R^{\zeta \times n}, \\ \overline{\Gamma}_\zeta &= \text{diag}\{\Gamma_\zeta, \dots, \Gamma_\zeta\} \in R^{\zeta \times \zeta}, \\ \overline{S}_\zeta &= \text{diag}\{S_\zeta, \dots, S_\zeta\} \in R^{\zeta \times n}. \end{aligned} \quad (41)$$

□

It is worth pointing out that $\overline{\eta}_\zeta/M_s$ and $-\sigma \Gamma_\zeta \widehat{W}_{\zeta i}$ are small perturbations by choosing large enough M_s and small enough σ . Therefore, the system (40) can be regarded as a class of LTV systems with very small perturbations. It has been shown in [28] that the nominal part of the system (40)

can be guaranteed to be exponentially stable if the system (40) satisfies the following three conditions:

- (i) There exists a positive constant ψ such that, for all $t \geq 0$, the bound $\max\{\|B(t)\|, \|dB(t)/dt\|\} \leq \psi$ is satisfied;
- (ii) There exist symmetric and positive matrices $P(t)$ and $Q(t)$ such that $A^T(t)P(t) + P(t)A(t) + \dot{P}(t) = -Q(t)$;
- (iii) $S_\zeta(Z)$ satisfies the PE condition.

From Section 3, all closed-loop signals remain uniformly ultimately bounded. Therefore, the satisfaction of condition (i) can be easily checked. Moreover, we have verified that $S_\zeta(Z)$ satisfy the PE condition (see the above analysis of Theorem 9 for the details). Next, choose a matrix $P(t) = Y_2 G$. Since Y_2 and G are positive definite and symmetric, the matrix $P(t)$ is also symmetric and positive. Then, we have

$$\begin{aligned} A^T(t)P(t) + P(t)A(t) + \dot{P}(t) \\ = 2[-K_2(t)G + \dot{G}^{-1}G]Y_2G + \dot{Y}_2G + Y_2\dot{G}. \end{aligned} \quad (42)$$

The inequality $A^T P + PA + \dot{P} < 0$ holds when choosing the appropriate control parameter $K_2(t) > (2\dot{G}^{-1}GY_2G + \dot{Y}_2G + Y_2\dot{G}) * (GY_2G)^{-1}/2$. Therefore, we has verified that the nominal part of the system (40) is uniformly exponentially stable. Further, based on the perturbation theory given in Lemma 4.6 [56], the weight estimate error $\widehat{W}_{i\zeta}$ converges exponentially to a small neighborhood of zero for $\forall t \geq T$. Noting that $\widehat{W}_{i\zeta} = \widehat{W}_{i\zeta} - W_{i\zeta}^*$, so $\widehat{W}_{i\zeta}$ converges exponentially to a small neighborhood of the desired weight value $W_{i\zeta}^*$ in a finite time T , and the corresponding desired weight value can be stored by constant neural weight values \overline{W}_i in (39).

Based on the spatially localized approximation property of RBF NN (6) and the constant weight values \overline{W} , the unknown system dynamics $\phi(Z)$ could be expressed as

$$\phi(Z) = \overline{W}_\zeta^T S_\zeta(Z) + \overline{\eta}_\zeta = \overline{W}^T S(Z) + \overline{\eta}, \quad (43)$$

where $\overline{\eta}_\zeta$ and $\overline{\eta}$ are close to η . From (43), neural networks $\overline{W}^T S(Z)$, containing the experience knowledge \overline{W} , can be used to accurately approximate the unknown system dynamics $\Phi(Z)$.

Furthermore, the learned knowledge can be described as follows: for the experienced recurrent orbit $\varphi_d(x_d(t))$, there exist positive constants d and ζ^* , which describe a local region Ω_{x_d} along $\varphi_d(x_d(t))$ such that

$$\begin{aligned} \text{dist}(\varphi(Z(t))|_{t \geq T}, \varphi_d(x_d(t))) < d \implies \\ \left| \overline{W}^T S(Z) - \phi(Z) \right| < \zeta^*, \end{aligned} \quad (44)$$

where ζ^* is close to η^* . For a new task, once the NN inputs $Z(t)$ enter the region Ω_{x_d} , the trained RBF networks $\overline{W}^T S(Z)$ can accurately approximate the uncertain nonlinearity $\phi(Z)$.

4.2. Static Controller Design with Knowledge Utilization. By invoking the stored weight values \overline{W} (39), a static controller

will be developed in this section to guarantee the prescribed performance of full-state tracking errors of the robotic system (1) for the same or a similar control task. Using the stored knowledge \bar{W} in (39), a static control law without neural weight estimated parameter adjustment online, instead of the adaptive NN control law (22), is designed as follows:

$$\tau = -k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 - \bar{W}^T S(Z), \quad (45)$$

where k_{20} and k_{21} are positive design parameters and ξ_2 and Y_2 are defined in (12). Moreover, the virtual control law α is chosen the as same as Section 3; see (15) for the detail. Then, by combining (16), (19), and (45), we can obtain the following closed-loop system:

$$\begin{aligned} \dot{\xi}_1 &= Y_1(e_2 - k_1\xi_1), \\ \dot{\xi}_2 &= Y_2M^{-1} \left[-k_{20}Y_2^{-1}\xi_2 - k_{21}Y_2\xi_2 + \Phi(Z) \right. \\ &\quad \left. - \bar{W}^T S(Z) - Y_2^{-1}V_m(q, \dot{q})\xi_2 + \eta' \right], \end{aligned} \quad (46)$$

where $\eta' = -d(t)$. Subsequently, construct the following Lyapunov function candidate:

$$\begin{aligned} V_1 &= \frac{\xi_1^T \xi_1}{2}, \\ V_2 &= \frac{1}{2} \xi_2^T M \xi_2. \end{aligned} \quad (47)$$

Noting the condition (44) and applying the similar backstepping step in Section 3, we have the following results.

Theorem 10. Consider the closed-loop system consisting of the robotic system (1), the bounded reference trajectory (2), the full-state tracking performance condition (3), the transformed error (10), the static neural learning control law (45) with the stored constant weight \bar{W} given in (39), and the virtual control law (15). Then, for the same or a similar recurrent reference orbit $\varphi_d(x_d(t))$ given in Theorem 6 and the initial conditions satisfying the prescribed performance (3), it can be guaranteed that all the closed-loop signals are uniformly ultimately bounded, and the constrained full-state tracking errors e_{ij} satisfy the prescribed performance (3) and converge to a small residual set of zero.

5. Simulation Results

To demonstrate the effectiveness of the proposed dynamic learning scheme, we consider a 2-link robot manipulator which is shown in Figure 1. From the n -link rigid robotic system (1), $q = [q_1, q_2]^T$ denotes the angular position of each joint and $\tau = [\tau_1, \tau_2]^T$ is the actuator input applied at the manipulator joints, respectively. Based on the system (1), the dynamic parameters of a 2-link robot manipulator are given by

$$M(q) = \begin{bmatrix} M_{11} & M_{12} \\ M_{12} & M_{22} \end{bmatrix},$$

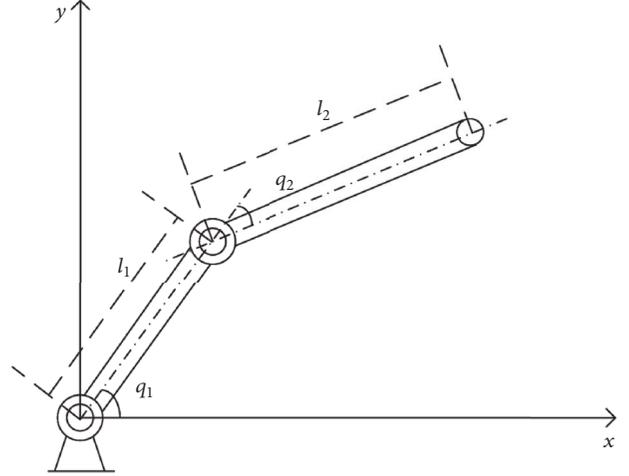


FIGURE 1: Diagram of a 2-link robotic manipulator.

$$F(\dot{q}) = \begin{bmatrix} 0.8\dot{q}_1 \\ 0.6\dot{q}_2 \end{bmatrix},$$

$$V_m(q, \dot{q})\dot{q} = \begin{bmatrix} -2l_1l_2m_2 \sin(q_2) (\dot{q}_1\dot{q}_2 + 0.5\dot{q}_2^2) \\ 0.52l_1l_2m_2 \sin(q_2) \dot{q}_1^2 \end{bmatrix},$$

$$G(q) = \begin{bmatrix} (m_1 + m_2)l_1g \cos(q_1) + m_2l_2g \cos(q_2) \\ m_2l_2g \cos(q_1 + q_2) \end{bmatrix}, \quad (48)$$

where

$$\begin{aligned} M_{11} &= l_2^2m_2 + l_1^2(m_1 + m_2) + 2l_1l_2m_2 \cos(q_2), \\ M_{12} &= l_2^2m_2 + 0.52l_1l_2m_2 \cos(q_2), \\ M_{22} &= l_2^2m_2, \end{aligned} \quad (49)$$

and l_i and m_i denote the length and the mass of link- i , $i = 1, 2$, and g denotes the gravity acceleration. In this paper, these system parameters are chosen as $m_1 = m_2 = 1$ kg, $l_1 = 0.8$ m, $l_2 = 2.3$ m, and $g = 9.8$ m/s², and the external disturbances $d = [0.1 \sin(t), 0.1 \cos(t)]^T$, which are bounded and satisfy Assumption 1.

The system output y is required to track the following desired reference trajectory $Y_d = [y_{d1}, y_{d2}]^T$:

$$\begin{aligned} y_{d1} &= \frac{80\pi}{360} \sin\left(\frac{2\pi t}{7}\right), \\ y_{d2} &= \frac{60\pi}{360} \cos\left(\frac{2\pi t}{7}\right). \end{aligned} \quad (50)$$

For full-state constrained tracking errors $e_1 = y - y_d = [e_{11}, e_{12}]^T$ and $e_2 = \dot{q} - \alpha_1 = [e_{21}, e_{22}]^T$, our target is to achieve the following prescribed transient and steady-state tracking error bounds:

$$-\underline{\delta}_{ij}\rho_i(t) < e_{ij}(t) < \bar{\delta}_{ij}\rho_i(t), \quad (51)$$

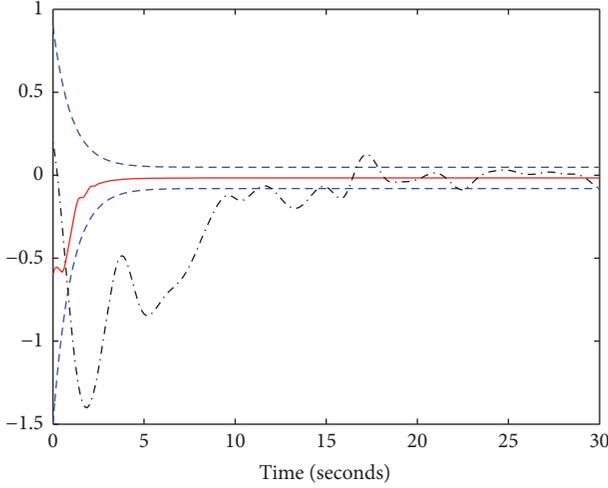


FIGURE 2: Angular position tracking error e_{11} : ANC with error transformation method (—), ANC in [30] (---), and the error bounds (- - -).

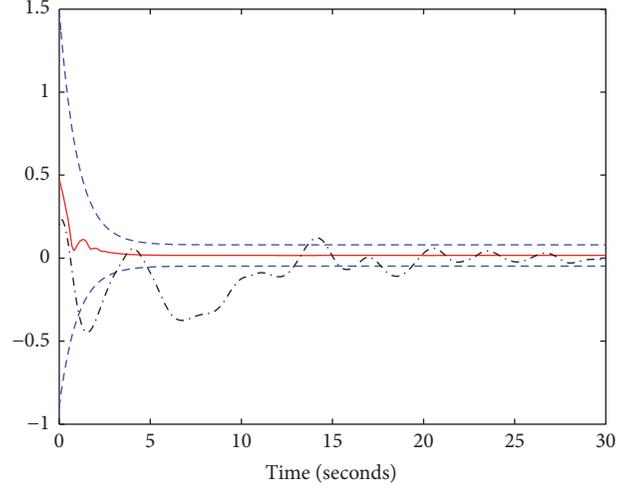


FIGURE 3: Angular position tracking error e_{12} : ANC with error transformation method (—), ANC method in [30] (---), and the error bounds (- - -).

where $i = 1, 2, j = 1, 2, \underline{\delta}_{11} = \underline{\delta}_{21} = 0.6, \bar{\delta}_{11} = \bar{\delta}_{21} = 1, \underline{\delta}_{12} = \underline{\delta}_{22} = 1$, and $\bar{\delta}_{12} = \bar{\delta}_{22} = 0.6$; the performance function $\rho_i(t)$ and the transformation function $T_{ij}^{-1}[e_{ij}/\rho_i(t)]$ are designed as

$$\rho_1(t) = (1.5 - 0.05)e^{-t} + 0.05, \quad \forall t \geq 0, \quad (52)$$

$$\rho_2(t) = (4 - 0.1)e^{-t} + 0.1, \quad \forall t \geq 0, \quad (53)$$

$$T_{ij}^{-1} \left[\frac{e_{ij}}{\rho_i(t)} \right] = \frac{1}{2} \ln \frac{\underline{\delta}_{ij} + e_{ij}/\rho_i(t)}{\bar{\delta}_{ij} - e_{ij}/\rho_i(t)}. \quad (54)$$

5.1. ANC Results with Full-State Tracking Error Constraints. According to Theorems 6 and 9, the main objective of this section is to use the proposed stable adaptive neural controller (22) with virtual control law (15) and neural weight adaptation law (23) such that the full-state tracking errors e_1 and e_2 satisfy the prescribed performance (51); the neural weight estimator \widehat{W} exponentially converges to the constant weight value \bar{W} ; the unknown system dynamics $\Phi(Z)$ in (20) is accurately approximated by the constant RBF NNs $\bar{W}^T S(Z)$.

In the simulation studies, the RBF network $W^T S(Z)$ consists of 3375 neurons whose centers are evenly spaced on $[-0.9, 0.9] \times [-0.9, 0.9] \times [-0.9, 0.9] \times [-0.5, 0.5] \times [-0.5, 0.5] \times [-0.5, 0.5]$ with the width $\eta_i = 0.4$. The other design parameters are chosen as follows: $k_{10} = 5, k_{20} = 25, k_{21} = 10, \Gamma = 15$, and $\sigma = 0.001$. The initial states are $\widehat{W}_1(0) = \widehat{W}_2(0) = 0, q(0) = [-0.6, 1]^T$, and $\dot{q}(0) = [1.5, -1]^T$. Simulation results are shown in Figures 2–10. Figures 2–5 show the constrained joint angular position and velocity tracking error performances, respectively. From Figures 2–5, it can be clearly seen that good transient performances have been achieved by adjusting the performance function (53) and design parameters $\underline{\delta}_{ij}, \bar{\delta}_{ij}$. The control input response is

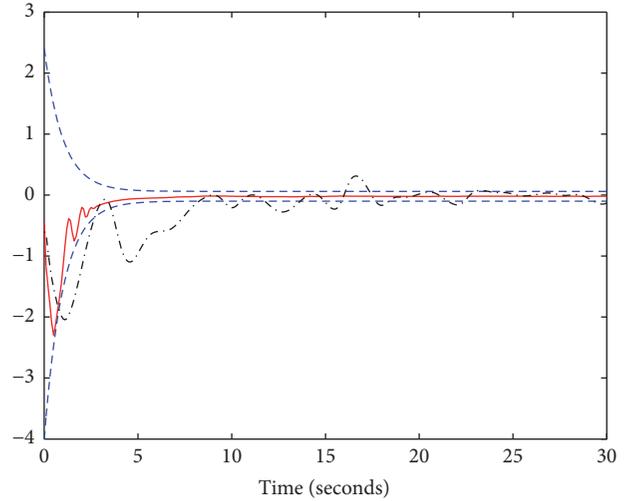


FIGURE 4: Angular velocity tracking error e_{21} : ANC with error transformation method (—), ANC in [30] (---), and the error bounds (- - -).

given in Figure 6. The partial weight convergence of $\widehat{W} = [\widehat{W}_1, \widehat{W}_2]$ is presented in Figures 7 and 8. It can be seen from Figures 7 and 8 that only partial weights converge to relatively large values, which means only the neurons along recurrent input signals Z can be motivated. Based on Theorem 9, the constant neural weight \bar{W}_i is chosen in the simulation as

$$\bar{W}_i = \min_{t \in [80, 100]} \widehat{W}_i(t), \quad i = 1, 2. \quad (55)$$

Figures 9 and 10 display unknown system dynamics $\Phi_1(Z)$ and $\Phi_2(Z)$, along the periodic reference signals Z , and can be accurately approximated by the constant RBF NN $\bar{W}_1^T S(Z)$ and $\bar{W}_2^T S(Z)$.

To further show the improved transient and steady-state tracking performance for the proposed control method with

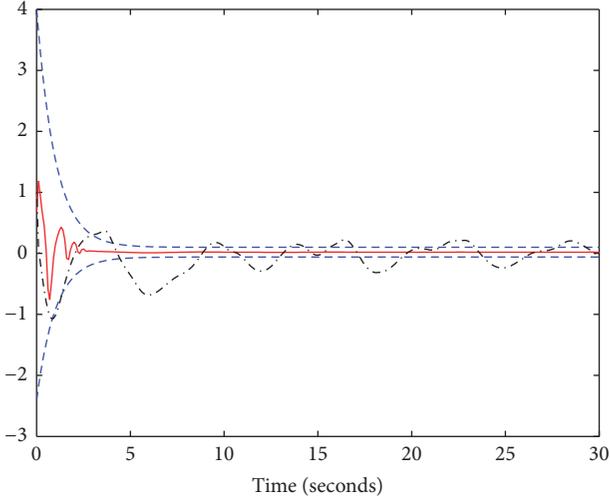


FIGURE 5: Angular velocity tracking error e_{22} : ANC with error transformation method (—), ANC method in [30] (---), and the error bounds (- - -).

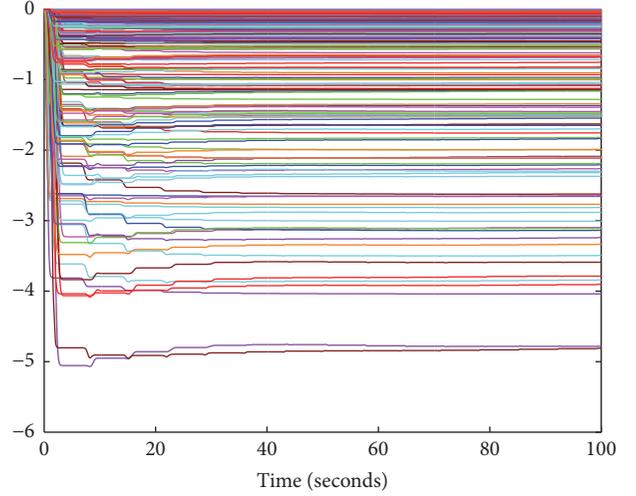


FIGURE 7: Partial parameter convergence of \widehat{W}_1 .

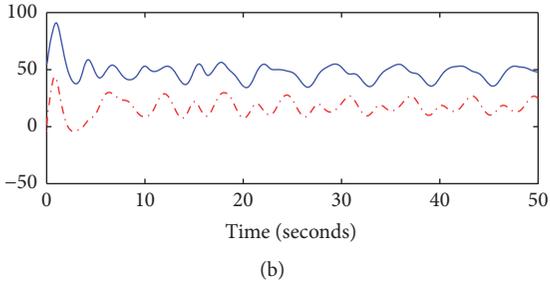
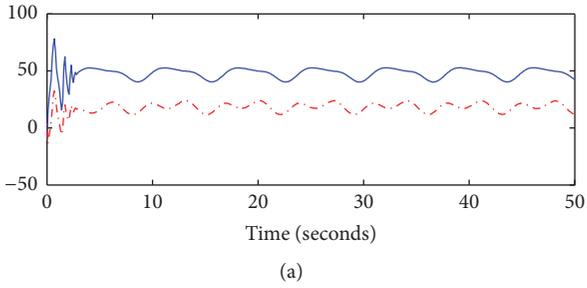


FIGURE 6: Control input responses: τ_1 (—), τ_2 (---): (a) ANC with error transformation and (b) ANC in [30].

full-state tracking performance constraints, the simulation comparison is given between the proposed method and the existing method without prescribed performances [30]. For comparison purpose, the existing method [30] is also used to control the same 2-link robot manipulator with the same initial condition $q(0) = [-0.6, 1]^T$, $\dot{q}(0) = [1.5, -1]^T$ and the same reference trajectory (50). For clarity, the existing method without prescribed performance proposed in [30] is recalled as follows: the control law is $\tau = -e_1 - c_2 e_2 - \widehat{W}^T S(Z)$ and neural weight updated laws $\dot{\widehat{W}} = \Gamma[S(Z)e_2^T - \sigma \widehat{W}]$ and $\alpha_1 = -c_1 e_1 + x_{d2}$. By choosing the appropriate control parameters $c_1 = 1$, $c_2 = 18$, $\sigma = 0.001$, and $\Gamma = 10$, to be fair, both control input signals of the two methods are

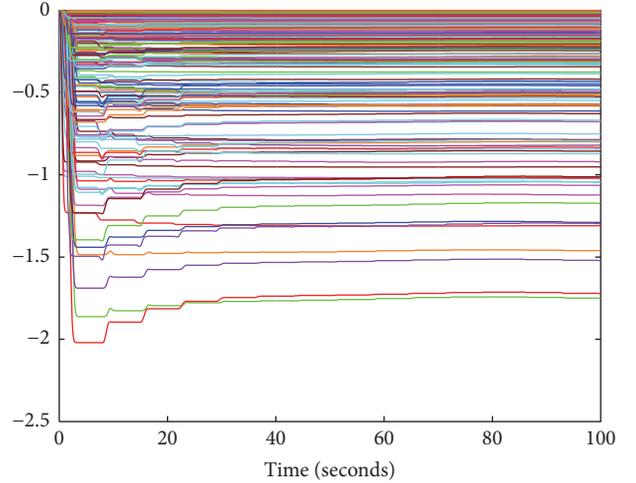


FIGURE 8: Partial parameter convergence of \widehat{W}_2 .

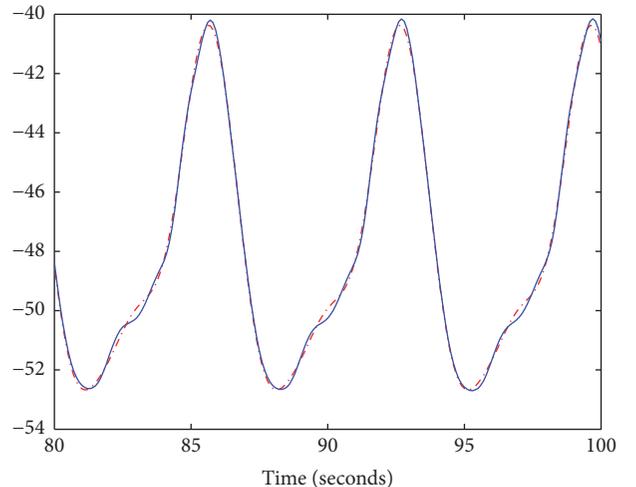


FIGURE 9: Function approximation: $\Phi_1(Z)$ (—) and $\widehat{W}_1^T S(Z)$ (---).

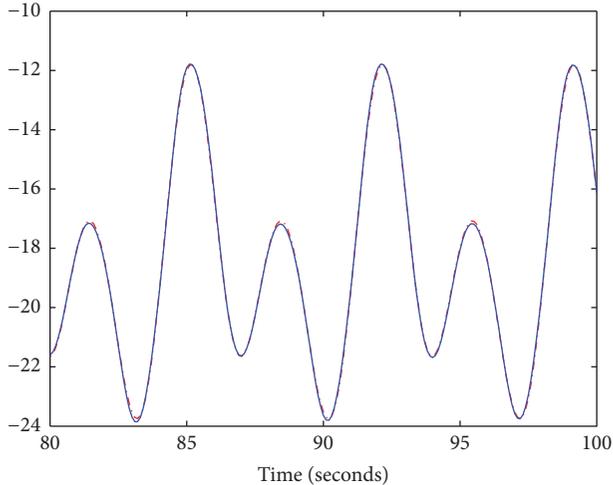


FIGURE 10: Function approximation: $\Phi_2(Z)$ (—) and $\overline{W}_2^T S(Z)$ (---).

required to have the similar amplitude, which is shown in Figure 6. The simulation results for comparison are given in Figures 2–5. It is clearly showed from Figures 2–5 that the proposed adaptive neural control scheme with full-state tracking performance constraints achieves a better transient and steady-state tracking control performance with smaller overshoot, faster convergence rate, and smaller steady-state error.

Remark 11. It is well known that the tracking performance relies on the choice of the control parameters and structure of RBF neural networks. However, how to choose design parameters for achieving the good tracking performance is still an open problem. In this simulation, the RBF networks $W^T S(Z)$ are constructed appropriately such that all neurons can cover the entire NN input trajectory $Z(t)$ space. Moreover, these controller parameters are chosen with large enough k_{10} , k_{20} , k_{21} , and Γ and small enough σ . It should be pointed out that these design parameters are chosen in this simulation by a trial-and-error method.

5.2. Neural Learning Control Results. By using the stored constant weight values \overline{W}_i in (55), the objective of this section is to invoke a static neural learning controller (45) to achieve the improved control performance with full-state tracking error constraints for the same or similar control tasks. For comparison purpose, the plant controlled, the reference trajectory, and the constrained tracking error performance are chosen the same as Section 5.1, while the initial conditions and the neural network structures are unchanged. In the simulation, with the control gains selected as $k_{10} = 3$, $k_{20} = 15$, and $k_{21} = 8$, simulation results for static neural learning control (45) are shown in Figures 11–15. From Figures 11–14, it can be seen that the smaller overshoot and the faster convergence are obtained using the learned knowledge \overline{W}_i in (55), while full-state tracking errors satisfy the prescribed performance. It is worth pointing out that a smaller control signal is used in neural learning control to achieve the

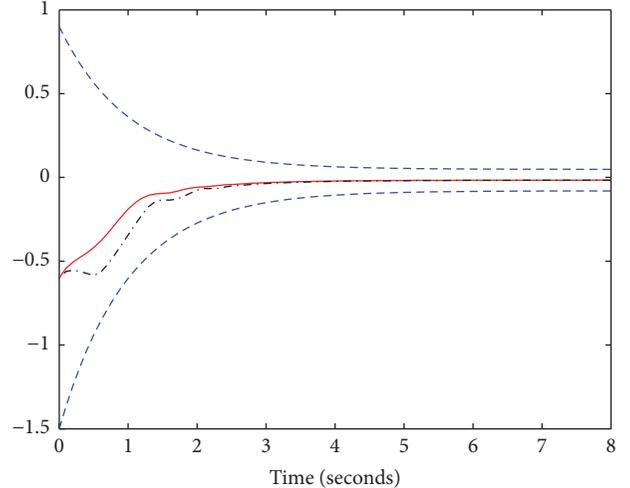


FIGURE 11: Angular position tracking error e_{11} : neural learning control (—), ANC (---), and the error bounds (- - -).

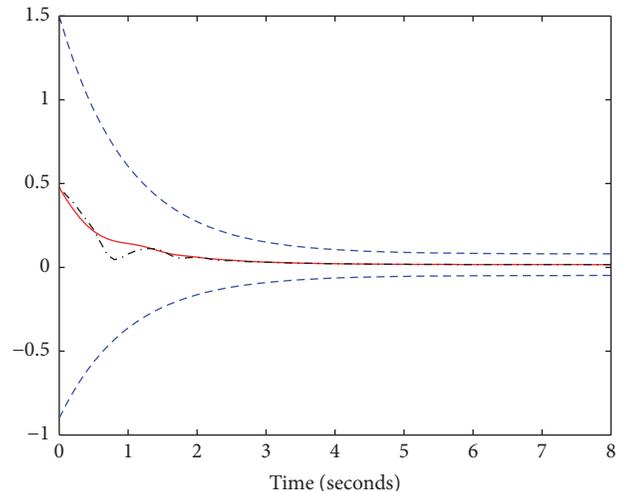


FIGURE 12: Angular position tracking error e_{12} : neural learning control (—), ANC (---), and the error bounds (- - -).

forementioned improved tracking control performance; see Figure 15 for the details. Moreover, because the proposed static learning control scheme avoids the online adjustment of the neural weight values, the running time saves nearly 1/2 for the same simulation time interval $t = [0, 100]$ s and the same computer configuration. The static neural learning controller (45) especially avoids the trial-and-error process on control design parameters and NN parameters which is tuned in adaptive neural control process. Therefore, the proposed learning control scheme avoids effectively a great deal of time consumed by the adaptive neural control process.

6. Conclusions

This paper focused on the problem of full-state tracking error constraints for an n -link rigid robot with unknown

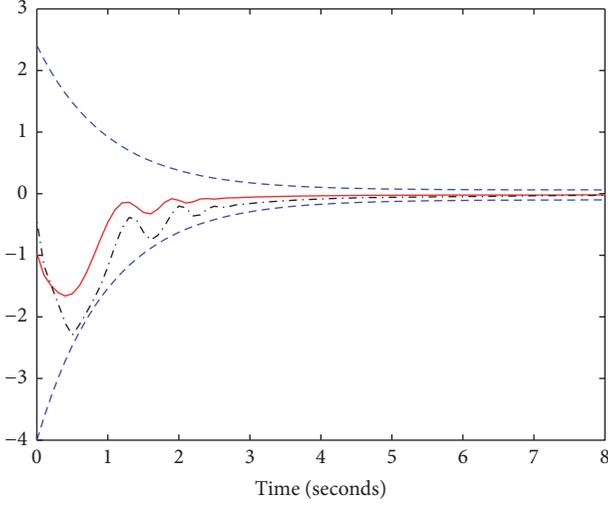


FIGURE 13: Angular velocity tracking error e_{21} : neural learning control (—), ANC (---), and the error bounds (- - -).

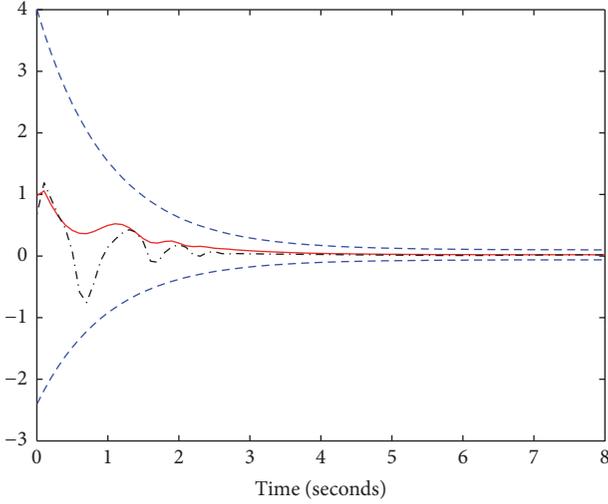


FIGURE 14: Angular velocity tracking error e_{22} : neural learning control (—), ANC (---), and the error bounds (- - -).

system dynamics and external disturbances. The performance transformation method was employed to transform the constrained full-state tracking errors into the unconstrained ones. By combining backstepping design and two independent Lyapunov functions, a novel adaptive neural control scheme was presented to guarantee all the signals in the closed-loop system are uniformly ultimately bounded, while this control scheme achieves predefined transient and steady-state tracking control performances concerning the link angular position and velocity tracking errors. Particularly, in the steady-state control process, the proposed neural control scheme can acquire, express, and store the knowledge of unknown system dynamics. The stored knowledge was reused to complete the same or similar tasks, so that the improved control performance was achieved with the less computational burden and the better transient-state tracking

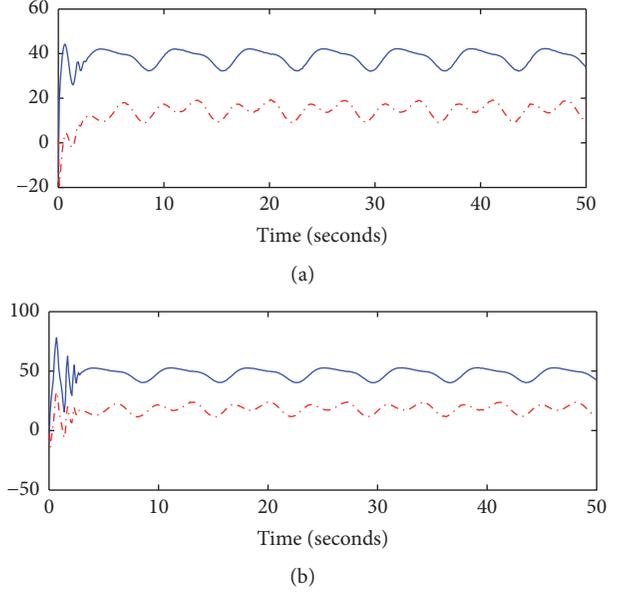


FIGURE 15: Control input responses: τ_1 (—), τ_2 (---): (a) neural learning control and (b) ANC.

performance. It should be pointed out that the considered n -link rigid robot is a class of simple multi-input and multioutput nonlinear systems. Therefore, how to extend the proposed method to complex nonlinear systems, such as nonaffine nonlinear systems, switched nonlinear systems, and stochastic large-scale systems, presents a challenging opportunity for future work.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (nos. 61374119, 61611130214, and 61473121), the Royal Society Newton Mobility Grant IE150858, the Guangdong Natural Science Foundation under Grant 2014A030312005, the Science and Technology New Star of Zhujiang, and the Fundamental Research Funds for the Central Universities.

References

- [1] F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators*, Macmillan, New York, NY, USA, 1993.
- [2] S. S. Ge, T. H. Lee, and C. J. Harris, *Adaptive Neural Network Control of Robotic Manipulators*, World Scientific, London, UK, 1998.
- [3] M. W. Spong and R. Ortega, "On adaptive inverse dynamics control of rigid robots," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 35, no. 1, pp. 92–95, 1990.

- [4] F. Cheng and G. Lee, "Robust control of manipulators using the computed torque plus H_∞ compensation method," *IEEE Proceedings - Control Theory and Applications*, vol. 143, no. 1, pp. 64–72, 1996.
- [5] Z. Li, J. Li, and Y. Kang, "Adaptive robust coordinated control of multiple mobile manipulators interacting with rigid environments," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 46, no. 12, pp. 2028–2034, 2010.
- [6] N. Sadegh and R. Horowitz, "Communications: An Exponentially Stable Adaptive Control Law For Robot Manipulators," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 491–496, 1990.
- [7] M. Namvar and F. Aghili, "Adaptive force-motion control of coordinated robots interacting with geometrically unknown environments," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 678–694, 2005.
- [8] D. Chwa, "Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 4, pp. 637–644, 2004.
- [9] J. Yang and J. Kim, "Sliding mode control for trajectory tracking of nonholonomic wheeled mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 3, pp. 578–587, 1999.
- [10] R. M. Sanner and J. E. Slotine, "Gaussian networks for direct adaptive control," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 837–863, 1992.
- [11] S. S. Ge and C. Wang, "Direct adaptive NN control of a class of nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 214–221, 2002.
- [12] Y.-J. Liu, L. Tang, S. Tong, and C. L. Chen, "Adaptive NN controller design for a class of nonlinear MIMO discrete-time systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 5, pp. 1007–1018, 2015.
- [13] M. Wang, X. Liu, and P. Shi, "Adaptive neural control of pure-feedback nonlinear time-delay systems via dynamic surface technique," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 41, no. 6, pp. 1681–1692, 2011.
- [14] T.-P. Zhang, H. Wen, and Q. Zhu, "Adaptive fuzzy control of nonlinear systems in pure feedback form based on input-to-state stability," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 80–93, 2010.
- [15] H. Q. Wang, P. Shi, H. Li, and Q. Zhou, "Adaptive neural tracking control for a class of nonlinear systems with unmodeled dynamics," *IEEE Transactions on Cybernetics*, 2016.
- [16] H. Wang, P. X. Liu, and P. Shi, "Observer-based fuzzy adaptive output-feedback control of stochastic nonlinear multiple time-delay systems," *IEEE Transactions on Cybernetics*, 2017.
- [17] S. Tong, Y. Li, and P. Shi, "Observer-based adaptive fuzzy backstepping output feedback control of uncertain MIMO pure-feedback nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 771–785, 2012.
- [18] C. M. Kwan, A. Yesildirek, and F. L. Lewis, "Robust force/motion control of constrained robots using neural network," *Journal of Robotic Systems*, vol. 16, no. 12, pp. 697–714, 1999.
- [19] Y. Karayiannidis, G. Rovithakis, and Z. Doulgeri, "Force/position tracking for a robotic manipulator in compliant contact with a surface using neuro-adaptive control," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 43, no. 7, pp. 1281–1288, 2007.
- [20] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2125–2136, 2017.
- [21] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.
- [22] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.
- [23] K. S. Fu, "Learning Control Systems and Intelligent Control Systems: An Intersection of Artificial Intelligence and Automatic Control," *IEEE Transactions on Automatic Control*, vol. 16, no. 1, pp. 70–72, 1971.
- [24] B. Xu, C. Yang, and Z. Shi, "Reinforcement learning output feedback NN control using deterministic learning technique," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 635–641, 2014.
- [25] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, 2017.
- [26] B. Luo, H.-N. Wu, and T. Huang, "Off-policy reinforcement learning for H_∞ control design," *IEEE Transactions on Cybernetics*, vol. 45, no. 1, pp. 65–76, 2015.
- [27] C. Wang and D. J. Hill, "Learning from neural control," *IEEE Transactions on Neural Networks*, vol. 17, no. 1, pp. 130–146, 2006.
- [28] T. Liu, C. Wang, and D. J. Hill, "Learning from neural control of nonlinear systems in normal form," *Systems & Control Letters*, vol. 58, no. 9, pp. 633–638, 2009.
- [29] S.-L. Dai, C. Wang, and M. Wang, "Dynamic learning from adaptive neural network control of a class of nonaffine nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 1, pp. 111–123, 2014.
- [30] C. Wang, M. Wang, T. Liu, and D. J. Hill, "Learning from ISS-modular adaptive NN control of nonlinear strict-feedback systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 10, pp. 1539–1550, 2012.
- [31] M. Wang and C. Wang, "Learning from adaptive neural dynamic surface control of strict-feedback systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 6, pp. 1247–1259, 2015.
- [32] S.-L. Dai, C. Wang, and F. Luo, "Identification and learning control of ocean surface ship using neural networks," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 4, pp. 801–810, 2012.
- [33] S.-L. Dai, M. Wang, and C. Wang, "Neural learning control of marine surface vessels with guaranteed transient tracking performance," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 3, pp. 1717–1727, 2016.
- [34] M. Wang and A. Yang, "Dynamic learning from adaptive neural control of robot manipulators with prescribed performance," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2244–2255, 2017.
- [35] W. He, A. O. David, Z. Yin, and C. Sun, "Neural network control of a robotic manipulator with input deadzone and output constraint," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 6, pp. 759–770, 2016.
- [36] K. P. Tee, S. S. Ge, and E. H. Tay, "Barrier Lyapunov functions for the control of output-constrained nonlinear systems," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 45, no. 4, pp. 918–927, 2009.

- [37] B. Ren, S. S. Ge, K. P. Tee, and T. H. Lee, "Adaptive neural control for output feedback nonlinear systems using a barrier Lyapunov function," *IEEE Transactions on Neural Networks*, vol. 21, no. 8, pp. 1339–1345, 2010.
- [38] W. He, C. Sun, and S. S. Ge, "Top tension control of a flexible marine riser by using integral-barrier Lyapunov function," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 497–505, 2015.
- [39] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [40] K. P. Tee and S. S. Ge, "Control of nonlinear systems with partial state constraints using a barrier Lyapunov function," *International Journal of Control*, vol. 84, no. 12, pp. 2008–2023, 2011.
- [41] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.
- [42] Y.-J. Liu and S. Tong, "Barrier Lyapunov functions-based adaptive control for a class of nonlinear pure-feedback systems with full state constraints," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 64, pp. 70–75, 2016.
- [43] C. P. Bechlioulis and G. A. Rovithakis, "Robust adaptive control of feedback linearizable MIMO nonlinear systems with prescribed performance," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 53, no. 9, pp. 2090–2099, 2008.
- [44] Y. Li, S. Tong, and T. Li, "Adaptive fuzzy output-feedback control for output constrained nonlinear systems in the presence of input saturation," *Fuzzy Sets and Systems. An International Journal in Information Science and Engineering*, vol. 248, pp. 138–155, 2014.
- [45] A. K. Kostarigka, Z. Doulgeri, and G. A. Rovithakis, "Prescribed performance tracking for flexible joint robots with unknown dynamics and variable elasticity," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 49, no. 5, pp. 1137–1147, 2013.
- [46] J. Na, Q. Chen, X. Ren, and Y. Guo, "Adaptive prescribed performance motion control of servo mechanisms with friction compensation," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 1, pp. 486–494, 2014.
- [47] Y. Li, Z. Ma, and S. Tong, "Adaptive fuzzy output-constrained fault-tolerant control of nonlinear stochastic large-scale systems with actuator faults," *IEEE Transactions on Cybernetics*, 2017.
- [48] Y. Li, S. Tong, L. Liu, and G. Feng, "Adaptive output-feedback control design with prescribed performance for switched nonlinear systems," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 80, pp. 225–231, 2017.
- [49] S. I. Han and J. M. Lee, "Partial tracking error constrained fuzzy dynamic surface control for a strict feedback nonlinear dynamic system," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 5, pp. 1049–1061, 2014.
- [50] S. Tong, S. Sui, and Y. Li, "Fuzzy adaptive output feedback control of MIMO nonlinear systems with partial tracking errors constrained," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 4, pp. 729–742, 2015.
- [51] C. P. Bechlioulis, Z. Doulgeri, and G. A. A. A. Rovithakis, "Neuro-adaptive force/position control with prescribed performance and guaranteed contact maintenance," *IEEE Transactions on Neural Networks*, vol. 21, no. 12, pp. 1857–1868, 2010.
- [52] M. Wang, C. Wang, and X. Liu, "Dynamic learning from adaptive neural control with predefined performance for a class of nonlinear systems," *Information Sciences. An International Journal*, vol. 279, pp. 874–888, 2014.
- [53] M. Wang, C. Wang, P. Shi, and X. Liu, "Dynamic learning from neural control for strict-feedback systems with guaranteed predefined performance," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 12, pp. 2564–2576, 2016.
- [54] R. Selmic and F. Lewis, "Deadzone compensation in motion control systems using neural networks," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 45, no. 4, pp. 602–613, 2000.
- [55] A. J. Kurdila, F. J. Narcowich, and J. D. Ward, "Persistency of excitation in identification using radial basis function approximations," *SIAM Journal on Control and Optimization*, vol. 33, no. 2, pp. 625–642, 1995.
- [56] H. K. Khalil, *Nonlinear Systems*, Prentice-Hall, Upper Saddle River, NJ, USA, 3rd edition, 2002.

Research Article

Composite Learning Sliding Mode Control of Flexible-Link Manipulator

Bin Xu and Pengchao Zhang

Shaanxi Provincial Key Laboratory of Industrial Automation, Shaanxi University of Technology, Hanzhong, Shaanxi 723000, China

Correspondence should be addressed to Bin Xu; smileface.binxu@gmail.com

Received 29 May 2017; Accepted 6 July 2017; Published 7 August 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Bin Xu and Pengchao Zhang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper studies the control of a flexible-link manipulator with uncertainty. The fast and slow dynamics are derived based on the singular perturbation (SP) theory. The sliding mode control is proposed while the adaptive design is developed using neural networks (NNs) and disturbance observer (DOB) where the novel update laws for NN and DOB are designed. The closed-loop system stability is guaranteed via Lyapunov analysis. The effectiveness of the proposed method is verified via simulation test.

1. Introduction

Flexible manipulators own the good characteristics of light weight, fast motion, and low energy consumption. Thus flexible manipulators can be used in many applications [1]. Due to the flexibility, the response of the manipulator shows oscillation while it is difficult to obtain high tracking precision. These are the two major challenges within the control of flexible manipulator [2]. As a result, many works have been aimed at controlling flexible manipulator [3–7].

In the literature, to deal with the dynamics transformation, output redefinition and SP method can be used. In [3], to avoid the difficulty of nonminimum phase system, the output is redefined for system transformation. In [8, 9], the observer based design is presented when the states are not available. In [10, 11], NNs are constructed to approximate the whole system uncertainty. The singularly perturbed model [12] is proposed to obtain fast dynamics and slow dynamics. Some other works can be found in [13, 14].

As discussed in [1], to achieve high tracking accuracy, based on output redefinition or SP method, efficient learning of system uncertainty and disturbance should be key factor. For system uncertainty, fuzzy logic system (FLS)/NNs can be employed. In the literature, there exist many results of intelligent control which employ intelligent system for approximation and then construct the controller [15–21]. One

concern is whether the FLS or NN has successfully fulfilled the task of approximation. To verify the effectiveness, the approximation error should be checked. However, usually it is not possible to derive the signal directly. In [1], with the output redefinition, the composite learning [22] is proposed with the serial-parallel estimation model. It is shown that the obtained predictor error can highly enhance the update of the learning system.

While the uncertainty commonly exists, disturbance might deteriorate the system tracking performance. With the upper bound for robust design, sliding mode control is studied. However, in this way, it brings energy consumption. One concern is to develop the efficient learning to follow the trend of the disturbance. The basic idea is that if the disturbance observer is fast enough compared with the system dynamics then to a great certain it can follow the trend of the disturbance. Some results can be referred to in [23, 24]. In [25], the attempt using composite learning with NN and DOB is developed for a flexible-link manipulator.

It is noted that in [25], the design is using backstepping scheme. To facilitate the design procedure, borrowing the idea of composite learning, the composite sliding mode control of n degrees of freedom flexible-link manipulators will be proposed.

The rest of the paper is arranged as follows. Section 2 presents the flexible-link manipulator dynamics and the

transformation with SP approach. The control of the slow subsystem and the fast subsystem is given in Sections 3 and 4, respectively. The simulation is shown in Section 5 while the conclusion is discussed in Section 6.

2. Dynamics Model

The model of n degrees of freedom flexible-link manipulators is as follows:

$$M \begin{bmatrix} \ddot{\theta} \\ \ddot{\delta} \end{bmatrix} + \begin{bmatrix} S_1(\theta, \delta, \dot{\theta}, \dot{\delta}) \\ S_2(\theta, \delta, \dot{\theta}, \dot{\delta}) \end{bmatrix} + \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\delta} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_2 \end{bmatrix} \begin{bmatrix} \theta \\ \delta \end{bmatrix} = \begin{bmatrix} u \\ 0 \end{bmatrix} + \begin{bmatrix} f_d \\ 0 \end{bmatrix}, \quad (1)$$

where the physical meanings of M , $S_1(\theta, \delta, \dot{\theta}, \dot{\delta})$, $S_2(\theta, \delta, \dot{\theta}, \dot{\delta})$, D_1 , D_2 , K_2 , u , and f_d can be found in [25].

With modal order as m , the vectors $\theta \in R^{n \times 1}$ and $\delta \in R^{mm \times 1}$ are defined as $\theta = [\theta_1, \dots, \theta_n]^T$ and $\delta = [\delta_{1,1}, \dots, \delta_{1,m}, \dots, \delta_{n,1}, \dots, \delta_{n,m}]^T$, where θ_i denotes the i -th joint angle variable and $\delta_{i,j}$ is the i -th link j -th modal variable.

Let $M^{-1} = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$ and denote $S_1(\theta, \delta, \dot{\theta}, \dot{\delta})$ and $S_2(\theta, \delta, \dot{\theta}, \dot{\delta})$ as S_1 and S_2 . Then the dynamics can be written as

$$\begin{aligned} \ddot{\theta} &= -H_{11} (S_1 + D_1 \dot{\theta} - u - f_d) \\ &\quad - H_{12} (S_2 + D_2 \dot{\delta} + K_2 \delta) \\ \ddot{\delta} &= -H_{21} (S_1 + D_1 \dot{\theta} - u - f_d) \\ &\quad - H_{22} (S_2 + D_2 \dot{\delta} + K_2 \delta). \end{aligned} \quad (2)$$

Define control input

$$u = u_s + u_f, \quad (3)$$

where u_s and u_f are the control inputs of the slow subsystem and the fast subsystem, respectively.

Supposing $k = \min K_2$, define $\rho = 1/k$, $\rho\phi = \delta$, and $\beta = \rho K_2$. The slow subsystem can be obtained as

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \zeta + bu_s + bf_{ds} = \zeta + b_0 u_s + d, \end{aligned} \quad (4)$$

where $x_1 = \theta_s$, $x_2 = \dot{\theta}_s$, $b = H_{11s} - H_{12s} H_{22s}^{-1} H_{21s}$, $\Delta b = b - b_0$, $\zeta = b(-S_{1s} - D_{1s} \dot{\theta}_s)$, and $d = bf_{ds} + \Delta b u_s$.

Remark 1 (b_0 is the nominal value of b). Define $\psi_1 = \phi - \phi_s$, $\psi_2 = \sqrt{\rho}\dot{\phi}$, and $\varsigma = t/\sqrt{\rho}$. The fast dynamics are written as

$$\begin{aligned} \frac{d\psi_1}{d\varsigma} &= \psi_2 \\ \frac{d\psi_2}{d\varsigma} &= -H_{22s}\beta\psi_1 - H_{22s}D_{2s}\sqrt{\rho}\psi_2 + H_{21s}u_f \end{aligned} \quad (5)$$

and the following expression is obtained:

$$\frac{d\psi}{d\varsigma} = P_f \psi + Q_f u_f, \quad (6)$$

where $\psi = [\psi_1, \psi_2]^T$, $P_f = \begin{bmatrix} 0 & I \\ -H_{22s}\beta & -H_{22s}D_{2s}\sqrt{\rho} \end{bmatrix}$, and $Q_f = [0, H_{21s}]^T$.

Remark 2. With the SP method in [12, 26], the fast dynamics (5) and the slow dynamics (4) are obtained. The detail to obtain (4) and (6) can be found in [25].

3. Composite Learning Sliding Mode Control of Slow Subsystem

Backstepping design is employed for controller design in [25] and during the analysis it will introduce the error signal in each step. In this paper, the sliding mode control will be proposed with composite learning design.

Define $e_1 = x_1 - x_{1d}$ and $e_2 = x_2 - \dot{x}_{1d}$, where x_{1d} and \dot{x}_{1d} are the desired joint angle trajectories.

Define the sliding surface

$$e_s = e_2 + ce_1, \quad (7)$$

where c is positive matrix.

Define $J = L_F \zeta$ and $x = [x_1^T, x_2^T]^T$, where L_F is a positive design constant. The following approximation exists:

$$\hat{J} = \hat{\omega}^T \vartheta(x), \quad (8)$$

where $\hat{\omega} \in R^{p \times n}$ is the weight matrix, p is the number of hidden nodes, and ϑ is the NN basis vector.

The derivative of e_s is calculated as

$$\begin{aligned} \dot{e}_s &= \dot{e}_2 + c\dot{e}_1 = \zeta + b_0 u_s + d - \ddot{x}_{1d} + c\dot{e}_1 \\ &= L_F^{-1} \hat{\omega}^T \vartheta(x) + b_0 u_s + L_F^{-1} \varepsilon + d - \ddot{x}_{1d} + c\dot{e}_1 \\ &= L_F^{-1} \hat{\omega}^T \vartheta(x) + b_0 u_s + F - \ddot{x}_{1d} + c\dot{e}_1, \end{aligned} \quad (9)$$

where $F = d + L_F^{-1} \varepsilon$ and $\hat{\omega}$ is the optimal weights matrix of function J approximation.

Define the prediction error as

$$\begin{aligned} Z_N &= x_2 - \hat{x}_2 \\ \hat{\eta}_2 &= L_F^{-1} \hat{\omega}^T \vartheta(x) + b_0 u_s + \hat{F} + k_p Z_N, \end{aligned} \quad (10)$$

where $k_p > 0$ is the design constant and \hat{F} is adaptive signal constructed as

$$\begin{aligned} \hat{F} &= L(x_2 - z) \\ \dot{z} &= L_F^{-1} \hat{\omega}^T \vartheta(x) + b_0 u_s + \hat{F} - L^{-1}(\gamma_z Z_N + e_s) \end{aligned} \quad (11)$$

with $L > 0$ and $\gamma_z > 0$ as design parameters.

Define $\tilde{\omega} = \omega - \hat{\omega}$, $\tilde{F} = F - \hat{F}$ where $\hat{\omega}$ is the estimation of ω and \hat{F} is the estimation of F . Then we have

$$\dot{\hat{F}} = L(\dot{x}_2 - \dot{z}) = L[L_F^{-1} \tilde{\omega}^T \vartheta(x) + \tilde{F}] + \gamma_z Z_N + e_s. \quad (12)$$

Finally u_s is proposed as

$$u_s = b_0^{-1} \left[-L_F^{-1} \tilde{\omega}^T \vartheta(x) - k_1 e_s - k_2 \text{sign}(e_s) - \tilde{f} + \ddot{x}_{1d} - c\dot{e}_1 \right] \quad (13)$$

where $k_1 \in R^{n \times n}$ and $k_2 \in R^{n \times n}$ are positive definite symmetric matrices.

The NN update law is proposed as

$$\dot{\hat{\omega}} = \gamma \left[\vartheta(x) L_F^{-1} (e_s + \gamma_z Z_N)^T - \sigma \hat{\omega} \right], \quad (14)$$

where $\gamma > 0$ and $\sigma > 0$ are design constants.

The error dynamics are obtained as

$$\dot{e}_s = L_F^{-1} \tilde{\omega}^T \vartheta - k_1 e_s - k_2 \text{sign}(e_s) + \tilde{f} \quad (15)$$

$$\dot{\tilde{f}} = \dot{f} - L \left[L_F^{-1} \tilde{\omega}^T \vartheta(x) + \tilde{f} \right] - \gamma_z Z_N - e_s \quad (16)$$

$$\dot{\hat{\omega}} = -\gamma \left[\vartheta(x) L_F^{-1} (e_s + \gamma_z Z_N)^T - \sigma \hat{\omega} \right] \quad (17)$$

and the derivative of Z_N is obtained as

$$\dot{Z}_N = L_F^{-1} \tilde{\omega}^T \vartheta + \tilde{f} - k_p Z_N. \quad (18)$$

Theorem 3. *With the controller (13), NN update law (14), and nonlinear DOB design (11), then all the signals in (A.1) are bounded.*

See appendix for proof.

4. Sliding Mode Control of Fast Subsystem

The control input u_f of the fast subsystem is designed as [25]

$$u_f = - \left(GQ_f \right)^+ \left[GP_f \psi + K_f \text{sat}(\xi) \right], \quad (19)$$

where K_f is the positive definite gain matrix.

The control input is presented as

$$u = u_s + u_f = b_0^{-1} \left[-L_F^{-1} \tilde{\omega}^T \vartheta(x) - k_1 e_s - k_2 \text{sat}(e_s) - \tilde{f} + \ddot{x}_{1d} - c\dot{e}_1 \right] - \left(GQ_f \right)^+ \left[GP_f \psi + K_f \text{sat}(\xi) \right]. \quad (20)$$

5. Simulation Example

To verify the effectiveness of the proposed method, simulation of the 2-DOF flexible-link manipulators is given. The parameter selection is selected as the same as [25]. The reference signals are given as

$$\begin{aligned} x_{1d}(1) &= \sin(0.3\pi t) \\ x_{1d}(2) &= -\cos(0.3\pi t). \end{aligned} \quad (21)$$

The approach in this paper is marked as ‘‘CL-SMC’’ which means composite learning sliding mode control while the design using tracking error to update NN is denoted as ‘‘NN-SMC.’’

The control parameters are set as $k_1 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$, $k_2 = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$, $k_p = 0.5$, $\gamma = 10$, $\gamma_z = 100$, $\sigma = 1$, $L = 10$, $\rho = 0.02$,

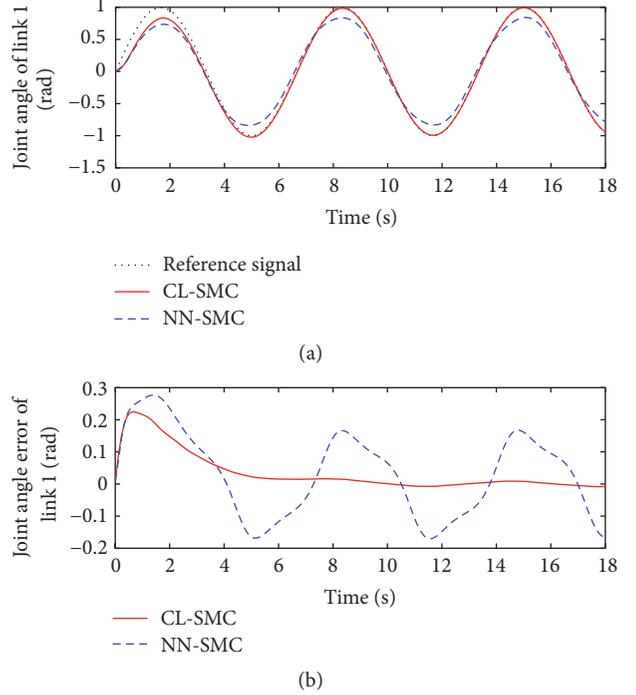


FIGURE 1: Joint angle of link 1. (a) System response. (b) Tracking error.

$K_f = 0.5I_{4 \times 4}$, $L_F = 2$, $G = [0.1I_{4 \times 4}, I_{4 \times 4}]$, $f_d(1) = 0.5 \sin t$, $f_d(2) = 0.5 \sin t$, and $c = 0.5I_{2 \times 2}$. The system tracking of link 1 and link 2 is shown in Figures 1 and 2, respectively. It is observed that under ‘‘CL-SMC,’’ much higher tracking accuracy can be obtained. Also the steady error is small for ‘‘CL-SMC’’ while for ‘‘NN-SMC’’ the error is large and chattering all the time. From Figures 3 and 4, the composite learning can closely follow the compound uncertainty while under ‘‘NN-SMC’’ the NN cannot fulfill the task. It confirms the rationales using composite learning. The responses of NN weights, control inputs, and sliding mode surface are shown in Figures 5, 6, and 7, respectively. The signal e_s is converging the small neighborhood of zero.

6. Conclusion

Considering the flexible-link manipulators, this paper proposed the sliding mode control with NN and DOB for compound estimation. The composite learning control scheme can greatly enhance the tracking performance. The simulation results confirms the design philosophy that the composite learning can efficiently fulfil the estimation task.

Appendix

Proof. The Lyapunov candidate is chosen as

$$V = V_1 + V_2 + V_3 + V_4, \quad (A.1)$$

where $V_1 = (1/2)e_s^T e_s$, $V_2 = (1/2)\tilde{f}^T \tilde{f}$, $V_3 = (1/2\gamma)\text{tr}(\tilde{\omega}^T \tilde{\omega})$, and $V_4 = (1/2)\gamma_z Z_N^T Z_N$.

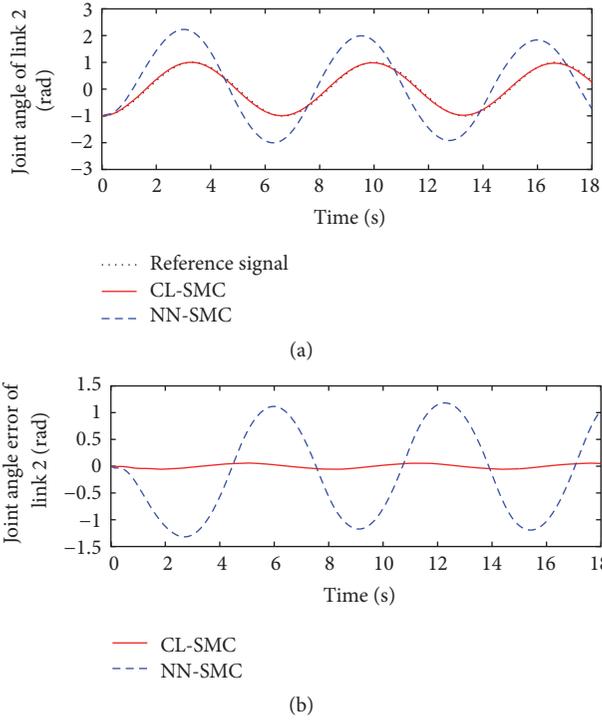


FIGURE 2: Joint angle of link 2. (a) System response. (b) Tracking error.

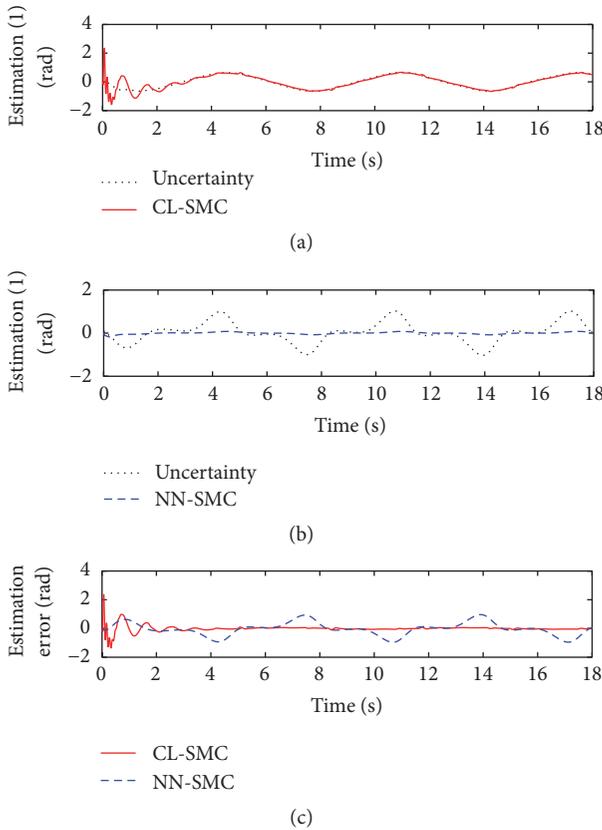


FIGURE 3: Uncertainty estimation 1. (a) Estimation under “CL-SMC.” (b) Estimation under “NN-SMC.” (c) Estimation error.

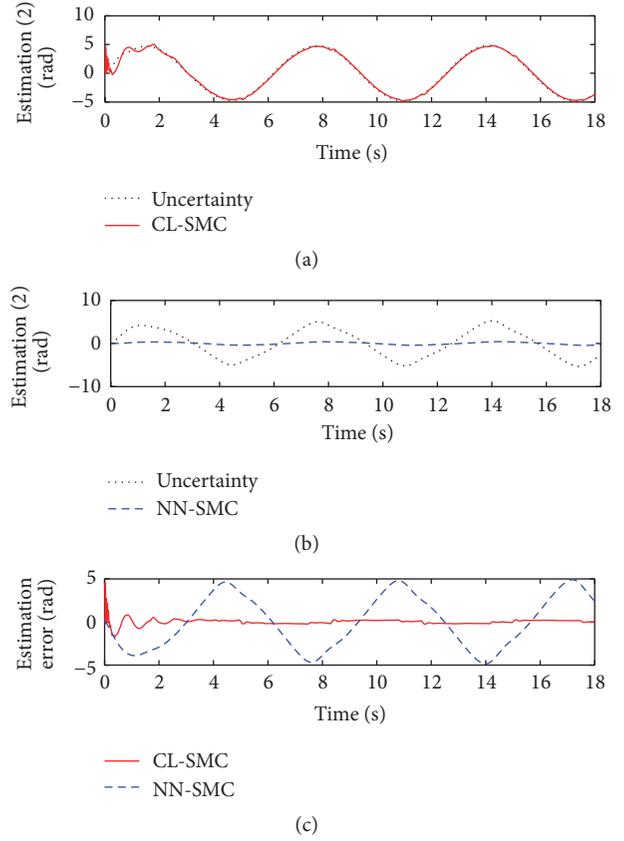


FIGURE 4: Uncertainty estimation 2. (a) Estimation under “CL-SMC.” (b) Estimation under “NN-SMC.” (c) Estimation error.

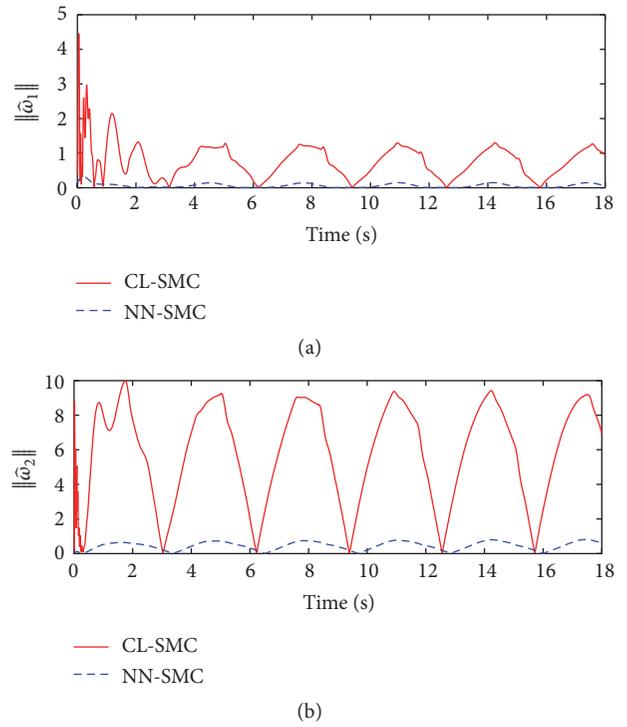
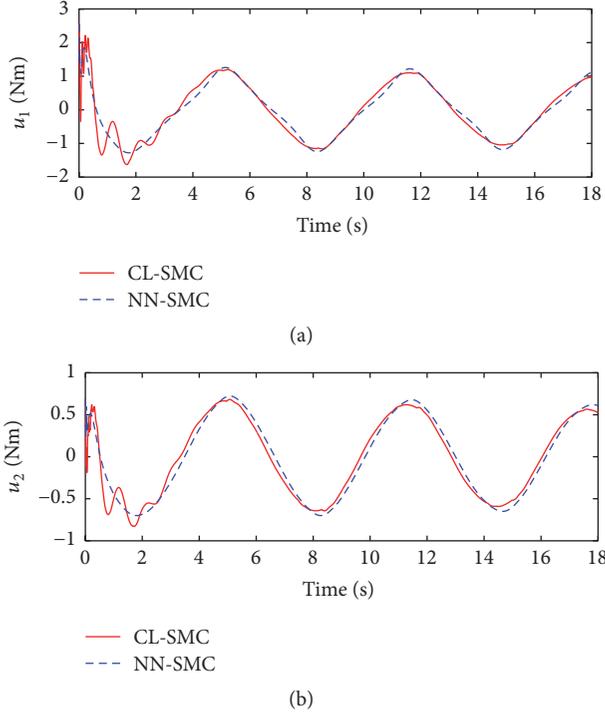


FIGURE 5: NN response. (a) Trajectory of $\|\hat{\omega}_1\|$. (b) Trajectory of $\|\hat{\omega}_2\|$.

FIGURE 6: Control signals. (a) u_1 . (b) u_2 .

Using (15), (17), (18), and (16), the derivatives of V_i , $i = 1, \dots, 4$ can be obtained as

$$\begin{aligned}
 \dot{V}_1 &= e_s^T \dot{e}_s \\
 &= e_s^T L_F^{-1} \bar{\omega}^T \vartheta - e_s^T k_1 e_s - e_s^T k_2 \text{sign}(e_s) + e_s^T \tilde{F} \\
 \dot{V}_2 &= \tilde{F}^T \dot{\tilde{F}} \\
 &= \tilde{F}^T \dot{\tilde{F}} - \gamma_z Z_N^T \tilde{F} - e_s^T \tilde{F} - L_F^{-1} L_F^T \bar{\omega}^T \vartheta - L_F^T \tilde{F} \\
 \dot{V}_3 &= \frac{1}{\gamma} \text{tr}(\bar{\omega}^T \dot{\bar{\omega}}) \\
 &= -L_F^{-1} e_s^T \bar{\omega}^T \vartheta - L_F^{-1} \gamma_z Z_N^T \bar{\omega}^T \vartheta \\
 &\quad - \sigma \text{tr}(\bar{\omega}^T \bar{\omega} - \bar{\omega}^T \omega) \\
 \dot{V}_4 &= \gamma_z Z_N^T \dot{Z}_N \\
 &= L_F^{-1} \gamma_z Z_N^T \bar{\omega}^T \vartheta + \gamma_z Z_N^T \tilde{F} - k_p \gamma_z Z_N^T Z_N.
 \end{aligned} \tag{A.2}$$

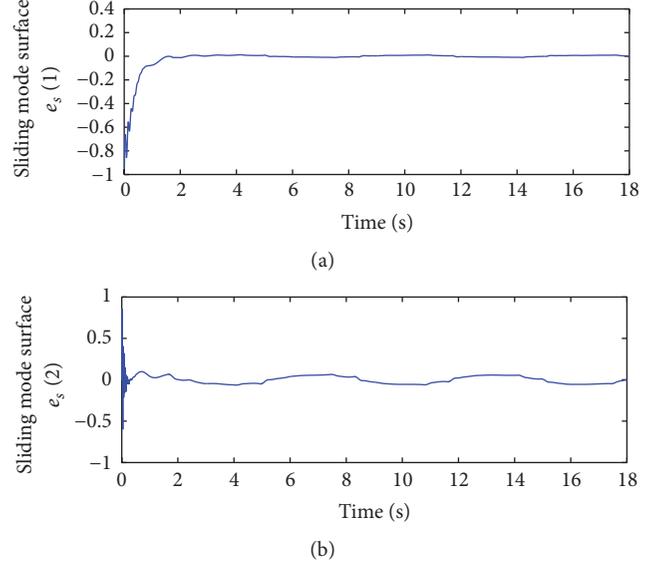
Then the derivative of V is calculated as

$$\begin{aligned}
 \dot{V} &= -e_s^T k_1 e_s - e_s^T k_2 \text{sign}(e_s) - \sigma \text{tr}(\bar{\omega}^T \bar{\omega} - \bar{\omega}^T \omega) \\
 &\quad - \gamma_z k_p Z_N^T Z_N + \tilde{F}^T \dot{\tilde{F}} - L_F^{-1} L_F^T \bar{\omega}^T \vartheta - L_F^T \tilde{F}.
 \end{aligned} \tag{A.3}$$

The following inequalities exist:

$$\begin{aligned}
 \text{tr}(\bar{\omega}^T \omega) &\leq \frac{1}{2} \text{tr}(\bar{\omega}^T \bar{\omega}) + \frac{1}{2} \text{tr}(\omega^T \omega) \\
 \tilde{F}^T \dot{\tilde{F}} &\leq \frac{1}{2} v_1 \tilde{F}^T \tilde{F} + \frac{1}{2v_1} \dot{\tilde{F}}^T \dot{\tilde{F}} \\
 -\tilde{F}^T (\bar{\omega}^T \vartheta) &\leq \frac{1}{2} v_2 \epsilon \tilde{F}^T \tilde{F} + \frac{1}{2v_2} \text{tr}(\bar{\omega}^T \bar{\omega}),
 \end{aligned} \tag{A.4}$$

where $\|\vartheta\|^2 \leq \epsilon$ and v_1 and v_2 are positive scalars.

FIGURE 7: Sliding mode surface. (a) Signal $e_s(1)$. (b) Signal $e_s(2)$.

Then we have

$$\begin{aligned}
 \dot{V} &\leq -e_s^T k_1 e_s - e_s^T k_2 \text{sign}(e_s) - \gamma_z k_p Z_N^T Z_N \\
 &\quad - \frac{1}{2} \sigma \text{tr}(\bar{\omega}^T \bar{\omega}) + \frac{1}{2} \sigma \text{tr}(\omega^T \omega) + \frac{1}{2} v_1 \tilde{F}^T \tilde{F} \\
 &\quad + \frac{1}{2v_1} \dot{\tilde{F}}^T \dot{\tilde{F}} + \frac{L}{2L_F} v_2 \tilde{F}^T \tilde{F} + \frac{L}{2v_2 L_F} \text{tr}(\bar{\omega}^T \bar{\omega}) \\
 &\quad - L_F^T \tilde{F} \\
 &= -e_s^T k_1 e_s - e_s^T k_2 \text{sign}(e_s) - \gamma_z k_p Z_N^T Z_N \\
 &\quad - \left(L - \frac{1}{2} v_1 - \frac{L v_2 \epsilon}{2L_F} \right) \tilde{F}^T \tilde{F} \\
 &\quad - \left(\frac{1}{2} \sigma - \frac{L}{2v_2 L_F} \right) \text{tr}(\bar{\omega}^T \bar{\omega}) + \frac{1}{2} \sigma \text{tr}(\omega^T \omega) \\
 &\quad + \frac{1}{2v_1} \dot{\tilde{F}}^T \dot{\tilde{F}}.
 \end{aligned} \tag{A.5}$$

By selecting appropriate parameters L , σ , and L_F to satisfy $k_d > 0$, $k_s > 0$, where

$$k_d = L - \frac{1}{2} v_1 - \frac{L v_2 \epsilon}{2L_F} \tag{A.6}$$

$$k_s = \frac{1}{2} \sigma - \frac{L}{2v_2 L_F},$$

it is concluded that

$$\dot{V} \leq -\chi V + \varphi, \tag{A.7}$$

where $\chi = \min[2\lambda_{\min}(k_1), 2\gamma k_s, 2k_p, 2k_d]$, $\varphi = (1/2)\sigma \text{tr}(\omega^T \omega) + (1/2v_1)\dot{\tilde{F}}^T \dot{\tilde{F}}$.

Then

$$0 \leq V \leq \frac{\varphi}{\chi} + \left[V(0) - \frac{\varphi}{\chi} \right] e^{-\chi t}. \quad (\text{A.8})$$

It is concluded when $t \rightarrow \infty$, $V \rightarrow \varphi/\chi$ and the signals included in (A.1) are bounded. \square

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61622308), Aeronautical Science Foundation of China (2015ZA53003), and Natural Science Basic Research Plan in Shaanxi Province (2016KJXX-86).

References

- [1] B. Xu and Y. Yuan, "Two performance enhanced control of flexible-link manipulator with system uncertainty and disturbances," *Science China Information Sciences*, vol. 60, no. 5, 2017.
- [2] M. Bodur and M. E. Sezer, "Adaptive control of flexible multilink manipulators," *International Journal of Control*, vol. 58, no. 3, pp. 519–536, 1993.
- [3] M. Moallem, K. Khorasani, and R. V. Patel, "Inversion-based sliding control of a flexible-link manipulator," *International Journal of Control*, vol. 71, no. 3, pp. 477–490, 1998.
- [4] S. Ulrich, J. Z. Sasiadek, and I. Barkana, "Nonlinear adaptive output feedback control of flexible-joint space manipulators with joint stiffness uncertainties," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 6, pp. 1961–1975, 2014.
- [5] S. Ulrich, J. Z. Sasiadek, and I. Barkana, "Modeling and direct adaptive control of a flexible-joint manipulator," *Journal of Guidance, Control, and Dynamics*, vol. 35, no. 1, pp. 25–39, 2012.
- [6] F. M. Caswara and H. Unbehauen, "A neurofuzzy approach to the control of a flexible-link manipulator," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 6, pp. 932–944, 2002.
- [7] Z. Liu, J. Liu, and W. He, "Adaptive boundary control of a flexible manipulator with input saturation," *International Journal of Control*, vol. 89, no. 6, pp. 1191–1202, 2016.
- [8] M. Moallem, R. V. Patel, and K. Khorasani, "Nonlinear tip-position tracking control of a flexible-link manipulator: Theory and experiments," *Automatica*, vol. 37, no. 11, pp. 1825–1834, 2001.
- [9] J. de León-Morales, J. G. Alvarez-Leal, R. Castro-Linares, and J. Alvarez-Gallegos, "Control of a flexible joint robot manipulator via a non-linear control-observer scheme," *International Journal of Control*, vol. 74, no. 3, pp. 290–302, 2001.
- [10] Y.-G. Tang, F.-C. Sun, Z.-Q. Sun, and T.-L. Hu, "Tip position control of a flexible-link manipulator with neural networks," *International Journal of Control, Automation and Systems*, vol. 4, no. 3, pp. 308–317, 2006.
- [11] Y. Tang, F. Sun, and Z. Sun, "Neural network control of flexible-link manipulators using sliding mode," *Neurocomputing*, vol. 70, no. 1-3, pp. 288–295, 2006.
- [12] B. Siciliano and W. J. Book, "A singular perturbation approach to control of lightweight flexible manipulators," *International Journal of Robotics Research*, vol. 7, no. 4, pp. 79–90, 1988.
- [13] M. J. Yazdanpanah, K. Khorasani, and R. V. Patel, "Uncertainty compensation for a flexible-link manipulator using nonlinear h control," *International Journal of Control*, vol. 69, no. 6, pp. 753–771, 1998.
- [14] M. Salehi and G. Vossoughi, "Impedance control of flexible base mobile manipulator using singular perturbation method and sliding mode control law," *International Journal of Control, Automation, and Systems*, vol. 6, no. 5, pp. 677–688, 2008.
- [15] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Y. Su, "Haptic identification by elm-controlled uncertain manipulator," *IEEE Transactions on Systems Man & Cybernetics: Systems*.
- [16] B. Luo, T. Huang, H.-N. Wu, and X. Yang, "Data-driven H_∞ control for nonlinear distributed parameter systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 11, pp. 2949–2961, 2015.
- [17] B. Xu and P. Zhang, "Minimal-learning-parameter technique based adaptive neural sliding mode control of MEMS gyroscope," *Complexity*, Article ID 6019175, 8 pages, 2017.
- [18] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*.
- [19] W. Chen, S. Hua, and H. Zhang, "Consensus-based distributed cooperative learning from closed-loop neural control systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 2, pp. 331–345, 2015.
- [20] F. Sun, L. Li, H.-X. Li, and H. Liu, "Neuro-fuzzy dynamic-inversion-based adaptive control for robotic manipulators - discrete time case," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 3, pp. 1342–1351, 2007.
- [21] F.-C. Sun, H.-X. Li, and L. Li, "Robot discrete adaptive control based on dynamic inversion using dynamical neural networks," *Automatica*, vol. 38, no. 11, pp. 1977–1983, 2002.
- [22] B. Xu, Z. Shi, C. Yang, and F. Sun, "Composite neural dynamic surface control of a class of uncertain nonlinear systems in strict-feedback form," *IEEE Transactions on Cybernetics*, vol. 44, no. 12, pp. 2626–2634, 2014.
- [23] B. Xu, "Disturbance observer-based dynamic surface control of transport aircraft with continuous heavy cargo airdrop," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 1, pp. 161–170, 2016.
- [24] B. Xu, "Composite learning finite-time control with application to quadrotors," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.
- [25] B. Xu, "Composite learning control of flexible-link manipulator using NN and DOB," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.
- [26] V. Etxebarria, A. Sanz, and I. Lizarraga, "Control of a lightweight flexible robotic arm using sliding modes," *International Journal of Advanced Robotic Systems*, vol. 2, no. 2, pp. 103–110, 2005.

Research Article

Multisynchronization for Coupled Multistable Fractional-Order Neural Networks via Impulsive Control

Jin-E Zhang

Hubei Normal University, Hubei 435002, China

Correspondence should be addressed to Jin-E Zhang; zhang86021205@163.com

Received 1 April 2017; Accepted 5 July 2017; Published 7 August 2017

Academic Editor: Guang Li

Copyright © 2017 Jin-E Zhang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

We show that every subnetwork of a class of coupled fractional-order neural networks consisting of N identical subnetworks can have $(r + 1)^n$ locally Mittag-Leffler stable equilibria. In addition, we give some algebraic criteria for ascertaining the static multisynchronization of coupled fractional-order neural networks with fixed and switching topologies, respectively. The obtained theoretical results characterize multisynchronization feature for multistable control systems. Two numerical examples are given to verify the superiority of the proposed results.

1. Introduction

Fractional calculus has been drawing interest over the last decade due to its many applications [1–12]. In mathematics and theoretical physics, fractional calculus is a generalization of integer-order calculus that roots in classical analysis theory. For system modeling, logical mechanism of fractional calculus is more sufficient and more essential [7, 12]. Then the models involving fractional calculus have greater ability to describe some real phenomena. To go in with mathematical tool behind fractional calculus, we have more room to develop new-type fractional dynamical systems. Actually, long and short memory with power law in dynamic evolution process are usually governed by the so-called fractional dynamical systems [5–7]. Many studies have shown that fractional dynamical systems have an unlimited memory. On the contrary, the memory in integer-order dynamical systems is limited. Moreover, in the general case, the evolution characteristics of fractional dynamical systems cannot be deduced by conventional integer-order variation. Besides, some existing formulations of fractional variation also meets bottleneck: (1) unitarity of evolution operator and (2) temporal fractality. Developing theoretical framework for a breakthrough in analysis and synthesis of fractional dynamical systems is even more important.

Synchronization phenomenon within complex networks is one of the most intriguing and valuable issues [13–16]. Recently, as a special case of an online learning situation, synchronization of neurodynamic systems have been found to be useful in characterizing and validating the consistency of cooperation [1, 8, 9, 11, 12]. In addition, neural synchronization under setting control rules offers an appealing alternative to neuron architectures and information exchange mechanism during neural mutual learning [16]. However, dynamics of neural synchronization process are intricate. The rhythms of biological brain emerge via synchronization between individual firings by activity-dependent coupling. To get a broader understanding of neural synchronization, it is reasonable to take a closer look at the dynamics of synchronous activity. When neurodynamic systems possess multiple locally stable equilibria, what structure should we use to describe the synchronization manifolds? Moreover, for multistable neurodynamic systems, how to implement this new synchronization control? As far as we know, there is little study. In addition, Lyapunov method [1, 8], Razumikhin-type stability theory [9, 16], infinitesimal generator on analytic semigroup principle [12], adaptive control [17], and mathematical induction method [18] are not very good at estimating the multisynchronization effect. Therefore, for the

multisynchronization of control systems, there would still be a lot of room left.

It is found that impulsive control is very effective in a wide variety of applications for performance improvement of control process [17–31]. Ayati and Khaloozadeh [19] study the adaptive impulsive control to design an observer for nonlinear continuous systems. Chen et al. [20] address the delayed impulsive control for exponential stability of Takagi-Sugeno fuzzy systems. Chen et al. [21] show that complex networks can synchronize under the impulsive control policy. It is also demonstrated that second-order consensus can be achieved via impulsive control algorithms [23]. In [24], impulsive control scheme is utilized to improve the performance of differential evolution. In [26], by using the impulsive control method, the ultimate boundedness problem of nonautonomous complex networks is investigated. Li and Song [27] take full advantage of delay-dependent impulsive control to analyze the stabilization problem of time-delay systems. Liu and Zhang [29] establish the impulsive control principle for stabilization of discrete-time nonlinear systems. Together with comparison criterion, fuzzy impulsive control is used for stabilization of chaotic systems [30]. Impulsive control algorithms are designed for droop-based secondary distributed control in islanded microgrids [31]. In [17], by the impulsive control schemes, exponential synchronization of complex dynamical networks in the presence of stochastic perturbations is formulated. In [18], under delayed impulsive control, stochastic synchronization problem for complex dynamical networks is addressed. Nevertheless, it should be pointed out that the impulsive control for multistable complex systems is still in early stage.

In view of the above discussions, the main objective of this paper is to investigate the multisynchronization for coupled multistable fractional-order neural networks via impulsive control. Several sufficient conditions are obtained to ensure that every subnetwork of coupled fractional-order neural networks has $(2r + 1)^n$ equilibria and $(r + 1)^n$ equilibria are locally Mittag-Leffler stable. The initial-value-related static multisynchronization is introduced to characterize the synchronous behaviors of the controlled coupled multistable fractional-order neural networks. The main emphasis will be then on impulsive control strategy to guarantee multisynchronization for coupled multistable fractional-order neural networks with fixed/switching topology. It is also shown that multisynchronization manifolds can sustain and maintain high levels for long running process. The proposed impulsive control strategy has a number of benefits: (1) saving communication bandwidth; (2) reducing communication cost; (3) good execution performance.

The rest of this paper is organized as follows. In Section 2, preliminaries and problem formulation are given. In Section 3, main results are derived to ascertain multisynchronization for coupled multistable fractional-order neural networks with fixed/switching topology. In Section 4, two numerical examples are presented to show the effectiveness of the obtained results. In Section 5, concluding remarks are stated.

2. Preliminaries and Problem Formulation

2.1. Notations. Throughout this paper, ${}^C D_{t_0}^q(\cdot)$ denotes Caputo fractional derivative operator. $\mathbf{1}^N$ is the N -dimensional column vector with its elements equal to 1. I_k is the k -dimensional identity matrix. $\mathcal{A}_1 \otimes \mathcal{A}_2$ denotes the Kronecker product of matrices \mathcal{A}_1 and \mathcal{A}_2 . Matrix $\mathcal{A} < 0$ (> 0 ; ≤ 0 ; ≥ 0) denotes that \mathcal{A} is negative definite (positive definite, negative semidefinite, and positive semidefinite). \mathcal{A}^{-1} represents the inverse matrix of \mathcal{A} . For the vector norm or the matrix norm, $\|\cdot\|$ stands for the Euclidean norm.

2.2. Model. Consider a class of coupled fractional-order neural networks consisting of N identical subnetworks

$${}^C D_{t_0}^q x_i(t) = -Dx_i(t) + BF(x_i(t)) + u_i(t), \quad (1)$$

$$t \geq t_0 \geq 0, \quad i = 1, 2, \dots, N,$$

where fractional-order $0 < q < 1$, $x_i(t) \in \mathfrak{R}^n$ is the state, $D = \text{diag}(d_1, d_2, \dots, d_n)$ is self-feedback term with $d_k > 0$, $k = 1, 2, \dots, n$, $B = (b_{kl})_{n \times n}$ represents synaptic strength, $F(x_i(t)) = (f_1(x_{i1}(t)), f_2(x_{i2}(t)), \dots, f_n(x_{in}(t)))^T$ is the feedback function, and $u_i(t) \in \mathfrak{R}^n$ denotes the input. The initial value of (1) is given by $x_i(t_0) = (x_{i1}(t_0), x_{i2}(t_0), \dots, x_{in}(t_0))^T$.

Now, we start to make the following assumptions for (1).

(A1) The feedback functions $F(x_i(t)) = (f_1(x_{i1}(t)), f_2(x_{i2}(t)), \dots, f_n(x_{in}(t)))^T$, $i = 1, 2, \dots, N$, satisfy

$$\mathcal{P}_k \leq f_k(s) \leq \overline{\mathcal{P}}_k, \quad \text{for } \forall s \in \mathfrak{R}, \quad (2)$$

$$\mathcal{Q}_k^- \leq \frac{f_k(s_1) - f_k(s_2)}{s_1 - s_2} \leq \mathcal{Q}_k^+, \quad (3)$$

$$\text{for } \forall s_1, s_2 \in \mathfrak{R}, \quad s_1 \neq s_2,$$

where $\mathcal{P}_k, \overline{\mathcal{P}}_k, \mathcal{Q}_k^-,$ and \mathcal{Q}_k^+ , $k = 1, 2, \dots, n$, are constants.

(A2) $D - \tilde{B}$ is a nonsingular M -matrix, where

$$\tilde{B} = (\tilde{b}_{kl})_{n \times n} \quad (4)$$

with

$$\tilde{b}_{kk} = \max\{b_{kk}Q_k^-, b_{kk}Q_k^+\}, \quad (5)$$

$$\tilde{b}_{kl} = \max\{|b_{kl}| |Q_l^-|, |b_{kl}| |Q_l^+|\}, \quad k \neq l.$$

(A3) To divide \mathfrak{R} into $2r+1$ intervals (i.e., $\mathfrak{R} = (-\infty, q_j^1) \cup [q_j^1, p_j^1] \cup (p_j^1, q_j^2) \cup [q_j^2, p_j^2] \cup \dots \cup [q_j^r, p_j^r] \cup (p_j^r, +\infty)$), then

$$-d_j q_j^\ell + b_{jj} f_j(q_j^\ell) + \sum_{k=1, k \neq j}^n \min\{b_{jk} \mathcal{P}_k, b_{jk} \overline{\mathcal{P}}_k\} + u_j(t) > 0, \quad (6)$$

$$-d_j p_j^\ell + b_{jj} f_j(p_j^\ell) + \sum_{k=1, k \neq j}^n \max\{b_{jk} \mathcal{P}_k, b_{jk} \overline{\mathcal{P}}_k\} + u_j(t) < 0,$$

for $j = 1, 2, \dots, n$, $\ell = 1, 2, \dots, r$, where $-\infty \leq q_j^1 < p_j^1 < q_j^2 < p_j^2 < \dots < q_j^r < p_j^r \leq +\infty$.

Under (A1)–(A3), every subnetwork of system (1) is multistable in Mittag-Leffler sense.

Lemma 1. *Let (A1)–(A3) hold, and every subnetwork of system (1) has $(2r + 1)^n$ equilibria and $(r + 1)^n$ equilibria are locally Mittag-Leffler stable.*

Using standard arguments as Theorem 1 in [2], Lemma 1 can be proved.

2.3. Properties. In this subsection, we give necessary definition and lemmas.

Denote $S_1, S_2, \dots, S_{(r+1)^n}$ as $(r + 1)^n$ locally Mittag-Leffler stable equilibria of every subnetwork of system (1).

Definition 2. System (1) is said to achieve static multisynchronization if the following holds:

(1) For any initial value $x(t_0) = (x_1^T(t_0), x_2^T(t_0), \dots, x_N^T(t_0))^T$ of (1), where $x_i(t_0) = (x_{i1}(t_0), x_{i2}(t_0), \dots, x_{in}(t_0))^T$, $i = 1, 2, \dots, N$, there exists $S_\ell \in \mathfrak{R}^n$ such that $\lim_{t \rightarrow +\infty} x_i(t) = S_\ell$, $i \in \{1, 2, \dots, N\}$, $\ell \in \{1, 2, \dots, (r + 1)^n\}$.

(2) The synchronization manifolds $\mathbf{1}^N \otimes S_\ell$ and $\mathbf{1}^N \otimes S_{\bar{\ell}}$ starting from different initial values $x(t_0) = (x_1^T(t_0), x_2^T(t_0), \dots, x_N^T(t_0))^T$ and $\tilde{x}(t_0) = (\tilde{x}_1^T(t_0), \tilde{x}_2^T(t_0), \dots, \tilde{x}_N^T(t_0))^T$, respectively, satisfy the following: there exists $\varepsilon > 0$ such that

$$\forall \tilde{x}(t) \in \{x(t) : 0 < \|x(t) - \mathbf{1}^N \otimes S_\ell\| < \varepsilon, x(t) \in \mathfrak{R}^{Nn}, t \geq t_0\}, \quad (7)$$

where $\tilde{x}(t)$ is not a point on $\mathbf{1}^N \otimes S_{\bar{\ell}}$.

Lemma 3. *Linear matrix inequality*

$$\begin{bmatrix} \mathcal{A}_1 & \mathcal{A}_2 \\ \mathcal{A}_2^T & \mathcal{A}_3 \end{bmatrix} < 0 \quad (8)$$

is equivalent to

$$(1) \mathcal{A}_1 < 0 \text{ and } \mathcal{A}_3 - \mathcal{A}_2^T \mathcal{A}_1^{-1} \mathcal{A}_2 < 0,$$

or

$$(2) \mathcal{A}_3 < 0 \text{ and } \mathcal{A}_1 - \mathcal{A}_2 \mathcal{A}_3^{-1} \mathcal{A}_2^T < 0,$$

where $\mathcal{A}_1 = \mathcal{A}_1^T$ and $\mathcal{A}_3 = \mathcal{A}_3^T$.

Lemma 4. *Let $\mathcal{A}(t)$ be a continuous function on $[t_0, +\infty)$; if there exists constant κ such that*

$${}^C D_{t_0}^q \mathcal{A}(t) \leq \kappa \mathcal{A}(t), \quad t \geq t_0 \geq 0, \quad (9)$$

then

$$\mathcal{A}(t) \leq \mathcal{A}(t_0) E_q(\kappa(t - t_0)^q), \quad t \geq t_0 \geq 0, \quad (10)$$

where $0 < q < 1$, $E_q(\cdot)$ is one-parameter Mittag-Leffler function.

3. Main Results

When the coupled fractional-order neural networks (1) generate multistable behavior, finding out the related multisynchronization control strategy is a challenging issue. In the following, we introduce impulsive control for multisynchronization in (1).

3.1. Fixed Topology Case. For the coupled fractional-order neural networks (1) with fixed topology, the impulsive control design is given below:

$$u_i(t) = a \sum_{h=1}^{+\infty} \left(\sum_{j=1, j \neq i}^N c_{ij} [x_j(t) - x_i(t)] \right) \delta(t - t_h), \quad (11)$$

for $i = 1, 2, \dots, N$, where $a > 0$ is the coupled gain, c_{ij} denotes the element of the weighted adjacency matrix of digraph \mathcal{D} , digraph \mathcal{D} possesses a directed spanning tree, $\delta(\cdot)$ is the Dirac Delta function, and the impulsive time sequence $\{t_h\}_{h=1}^{+\infty}$ satisfies $0 < t_1 < t_2 < \dots < t_h < \dots$ and $\lim_{h \rightarrow \infty} t_h = +\infty$.

Combining with (1) and (11), we get

$${}^C D_{t_0}^q x(t) = -(I_N \otimes D)x(t) + (I_N \otimes B)F(x(t)), \quad t \neq t_h, \quad (12)$$

$$\Delta x(t) = -(aL \otimes I_n)x(t^-), \quad t = t_h,$$

$$t \geq t_0 \geq 0, \quad h = 1, 2, \dots,$$

where $x(t) = (x_1^T(t), x_2^T(t), \dots, x_N^T(t))^T$, $F(x(t)) = (F^T(x_1(t)), F^T(x_2(t)), \dots, F^T(x_N(t)))^T$, and $L = (\mathcal{L}_{ij})_{N \times N}$ is the Laplacian matrix associated with digraph \mathcal{D} .

Theorem 5. *Let (A1)–(A3) hold; for any given constant $\alpha_1 > 0$, if there exist constant $0 < \alpha_2 < 1$ and matrices $\mathcal{R} > 0$ and $\mathcal{W} = \text{diag}(\mathcal{W}_1, \mathcal{W}_2, \dots, \mathcal{W}_n) > 0$ such that*

$$\begin{bmatrix} -\alpha_2 I_{N-1} & (I_{N-1} - a\mathcal{S})^T \\ (I_{N-1} - a\mathcal{S}) & -I_{N-1} \end{bmatrix} < 0, \quad (13)$$

$$\begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{\alpha_2}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix} < 0, \quad (14)$$

where

$$\mathcal{S} = (\mathcal{L}_{ij} - \mathcal{L}_{Nj})_{(N-1) \times (N-1)},$$

$$\mathcal{M}_1 = \text{diag}(\mathcal{Q}_1^+ \mathcal{Q}_1^-, \mathcal{Q}_2^+ \mathcal{Q}_2^-, \dots, \mathcal{Q}_n^+ \mathcal{Q}_n^-), \quad (15)$$

$$\mathcal{M}_2 = \text{diag}\left(\frac{\mathcal{Q}_1^+ + \mathcal{Q}_1^-}{2}, \frac{\mathcal{Q}_2^+ + \mathcal{Q}_2^-}{2}, \dots, \frac{\mathcal{Q}_n^+ + \mathcal{Q}_n^-}{2}\right),$$

then, for the impulsive time sequence $\{t_h\}_{h=1}^{+\infty}$ admitting $\sup\{t_h - t_{h-1}\} \leq \alpha_1$, system (12) can achieve static multisynchronization.

Proof. From (A1)–(A3), according to Lemma 1, every subnetwork of system (1) has $(r+1)^n$ locally Mittag-Leffler stable equilibria $S_1, S_2, \dots, S_{(r+1)^n}$.

Let $\mathcal{Y}_i(t) = x_i(t) - S$, $i = 1, 2, \dots, N$, where $S \in \{S_h, h = 1, 2, \dots, (r+1)^n\}$, from (12), and then

$${}^C D_{t_0}^q \mathcal{Y}(t) = -(I_N \otimes D) \mathcal{Y}(t) + (I_N \otimes B) F(\mathcal{Y}(t)), \quad t \neq t_h, \quad (16)$$

$$\Delta \mathcal{Y}(t) = -(aL \otimes I_n) \mathcal{Y}(t^-), \quad t = t_h, \\ t \geq t_0 \geq 0, \quad h = 1, 2, \dots,$$

where $\mathcal{Y}(t) = (\mathcal{Y}_1^T(t), \mathcal{Y}_2^T(t), \dots, \mathcal{Y}_N^T(t))^T$ and $F(\mathcal{Y}(t)) = (F^T(\mathcal{Y}_1(t)), F^T(\mathcal{Y}_2(t)), \dots, F^T(\mathcal{Y}_N(t)))^T$.

Let $\mathcal{X}_i(t) = \mathcal{Y}_i(t) - \mathcal{Y}_N(t)$, $i = 1, 2, \dots, N-1$, and system (16) can be reformulated as

$${}^C D_{t_0}^q \mathcal{X}(t) = -(I_{N-1} \otimes D) \mathcal{X}(t) \\ + (I_{N-1} \otimes B) F(\mathcal{X}(t)), \quad t \neq t_h, \\ {}^C D_{t_0}^q \mathcal{Y}_N(t) = -D \mathcal{Y}_N(t) + B F(\mathcal{Y}_N(t)), \quad t \neq t_h, \quad (17)$$

$$\Delta \mathcal{X}(t) = -(a\mathcal{S} \otimes I_n) \mathcal{X}(t^-), \quad t = t_h,$$

$$\Delta \mathcal{Y}_N(t) = -(a\widehat{\mathcal{S}} \otimes I_n) \mathcal{Y}_N(t^-), \quad t = t_h, \\ t \geq t_0 \geq 0, \quad h = 1, 2, \dots,$$

where $\mathcal{X}(t) = (\mathcal{X}_1^T(t), \mathcal{X}_2^T(t), \dots, \mathcal{X}_{N-1}^T(t))^T$, $F(\mathcal{X}(t)) = (F^T(\mathcal{X}_1(t)), F^T(\mathcal{X}_2(t)), \dots, F^T(\mathcal{X}_{N-1}(t)))^T$, and $\widehat{\mathcal{S}} = (\mathcal{L}_{N1}, \mathcal{L}_{N2}, \dots, \mathcal{L}_{N(N-1)})$.

Obviously, for $i = 1, 2, \dots, N-1$,

$$\mathcal{X}_i(t) = \mathcal{Y}_i(t) - \mathcal{Y}_N(t) = (x_i(t) - S) - (x_N(t) - S) \\ = x_i(t) - x_N(t). \quad (18)$$

According to Lemma 3, (13) is equivalent to

$$(I_{N-1} - a\mathcal{S})^T (I_{N-1} - a\mathcal{S}) \leq \alpha_2 I_{N-1}. \quad (19)$$

Define a Lyapunov function

$$V(t) = \mathcal{X}^T(t) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t). \quad (20)$$

When $t = t_h$, we can obtain

$$V(t_h) = \mathcal{X}^T(t_h) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t_h) \\ = \mathcal{X}^T(t_h^-) [(I_{N-1} - a\mathcal{S}) \otimes I_n]^T (I_{N-1} \otimes \mathcal{R}) \\ \times [(I_{N-1} - a\mathcal{S}) \otimes I_n] \mathcal{X}(t_h^-) \leq \alpha_2 V(t_h^-). \quad (21)$$

On the other hand, by (3) in (A1), for any given $\mathcal{W} = \text{diag}(\mathcal{W}_1, \mathcal{W}_2, \dots, \mathcal{W}_n) > 0$, we have

$$0 \leq \sum_{j=1}^n \mathcal{W}_j [\mathcal{Q}_j^+ \mathcal{X}_j(t) - F(\mathcal{X}_j(t))] \\ \cdot [F(\mathcal{X}_j(t)) - \mathcal{Q}_j^- \mathcal{X}_j(t)] = -\mathcal{X}_i^T(t) \mathcal{W} \mathcal{M}_1 \mathcal{X}_i(t) \\ + 2\mathcal{X}_i^T(t) \mathcal{W} \mathcal{M}_2 F(\mathcal{X}_i(t)) - F^T(\mathcal{X}_i(t)) \\ \cdot \mathcal{W} F(\mathcal{X}_i(t)), \quad (22)$$

and thus

$$0 \leq -\mathcal{X}^T(t) I_{N-1} \otimes (\mathcal{W} \mathcal{M}_1) \mathcal{X}(t) + 2\mathcal{X}^T(t) I_{N-1} \\ \otimes (\mathcal{W} \mathcal{M}_2) F(\mathcal{X}(t)) - F^T(\mathcal{X}(t)) I_{N-1} \\ \otimes \mathcal{W} F(\mathcal{X}(t)). \quad (23)$$

By (14), it follows that

$$\begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{(\alpha_2 + \alpha_3)}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix} \\ < 0, \quad (24)$$

where constant $\alpha_3 \in (0, 1 - \alpha_2)$.

When $t \neq t_h$, we can get

$${}^C D_{t_0}^q V(t) \leq 2\mathcal{X}^T(t) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t) + 2\mathcal{X}^T(t) \\ \cdot (I_{N-1} \otimes \mathcal{R}) \\ \cdot [-(I_{N-1} \otimes D) \mathcal{X}(t) + (I_{N-1} \otimes B) F(\mathcal{X}(t))]. \quad (25)$$

Together with (23)–(25),

$${}^C D_{t_0}^q V(t) \leq \mathcal{H}^T(t) (I_{N-1} \otimes \chi) \mathcal{H}(t) \\ - \frac{(\alpha_2 + \alpha_3)}{\alpha_1} V(t) < -\frac{(\alpha_2 + \alpha_3)}{\alpha_1} V(t), \quad (26)$$

where

$$\mathcal{H}^T(t) = (\mathcal{X}^T(t), F^T(\mathcal{X}(t)))^T, \\ \chi \\ = \begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{(\alpha_2 + \alpha_3)}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix}. \quad (27)$$

According to Lemma 4,

$$V(t) \leq V(t_0) E_q \left(-\frac{(\alpha_2 + \alpha_3)}{\alpha_1} (t - t_0)^q \right), \quad (28)$$

$$t \geq t_0 \geq 0,$$

so $V(t) \rightarrow 0$ as $t \rightarrow +\infty$, which implies that, for any given initial value of (1), $x_1(t) = x_2(t) = \dots = x_N(t)$ as $t \rightarrow +\infty$, and, hence, system (12) can reach complete synchronization. \square

Moreover, consider the N th subnetwork of (12):

$$\begin{aligned} {}^C D_{t_0}^q x_N(t) &= -Dx_N(t) + BF(x_N(t)), \quad t \neq t_h, \\ \Delta x_N(t) &= \sum_{j=1}^{N-1} c_{Nj} (x_j(t^-) - x_N(t^-)), \quad t = t_h, \\ t &\geq t_0 \geq 0, \quad h = 1, 2, \dots \end{aligned} \quad (29)$$

Based on the above analysis, we have $\Delta x_N(t) = 0$ as $t \rightarrow +\infty$. Through Lemma 1, the N th subnetwork of (12) has $(r+1)^n$ locally Mittag-Leffler stable equilibria $S_1, S_2, \dots, S_{(r+1)^n}$. Therefore, it follows that $x_1(t) = x_2(t) = \dots = x_N(t) = S$ as $t \rightarrow +\infty$, $S \in \{S_h, \bar{h} = 1, 2, \dots, (r+1)^n\}$. To sum up, system (12) can achieve static multisynchronization.

3.2. Switching Topology Case. Without loss of generality, we introduce a switching signal $\rho(t) : [t_0, +\infty) \rightarrow \{1, 2, \dots, \mathcal{K}\}$ and \mathcal{K} digraphs indexed by digraph \mathcal{D}^1 , digraph \mathcal{D}^2, \dots , digraph $\mathcal{D}^{\mathcal{K}}$.

For the coupled fractional-order neural networks (1) with switching topology, the impulsive control design is given below:

$$\begin{aligned} u_i(t) &= a^{\rho(t)} \sum_{h=1}^{+\infty} \left(\sum_{j=1, j \neq i}^N c_{ij}^{\rho(t)} [x_j(t) - x_i(t)] \right) \delta(t - t_h), \end{aligned} \quad (30)$$

for $i = 1, 2, \dots, N$, where the switching signal $\rho(t) : [t_0, +\infty) \rightarrow \{1, 2, \dots, \mathcal{K}\}$, $a^{\rho(t)} > 0$ is the coupled gain, $c_{ij}^{\rho(t)}$ denotes the element of the weighted adjacency matrix of digraph $\mathcal{D}^{\rho(t)}$, digraph $\mathcal{D}^{\rho(t)}$ possesses a directed spanning tree, $\delta(\cdot)$ is the Dirac Delta function, and the impulsive time sequence $\{t_h\}_{h=1}^{+\infty}$ satisfies $0 < t_1 < t_2 < \dots < t_h < \dots$ and $\lim_{h \rightarrow \infty} t_h = +\infty$.

Combining with (1) and (30), we get

$$\begin{aligned} {}^C D_{t_0}^q x(t) &= -(I_N \otimes D)x(t) + (I_N \otimes B)F(x(t)), \\ \Delta x(t) &= -(a^{\rho(t)} L^{\rho(t)} \otimes I_n)x(t^-), \quad t = t_h, \\ t &\geq t_0 \geq 0, \quad h = 1, 2, \dots, \end{aligned} \quad (31)$$

where $x(t) = (x_1^T(t), x_2^T(t), \dots, x_N^T(t))^T$, $F(x(t)) = (F^T(x_1(t)), F^T(x_2(t)), \dots, F^T(x_N(t)))^T$, and $L^{\rho(t)} = (\mathcal{L}_{ij}^{\rho(t)})_{N \times N}$ is the Laplacian matrix associated with digraph $\mathcal{D}^{\rho(t)}$.

Theorem 6. Let (A1)–(A3) hold; for any given constant $\alpha_1 > 0$, if there exist constant $0 < \alpha_2 < 1$ and matrices $\mathcal{R} > 0$ and $\mathcal{W} = \text{diag}(\mathcal{W}_1, \mathcal{W}_2, \dots, \mathcal{W}_n) > 0$ such that

$$\begin{bmatrix} -\alpha_2 I_{N-1} & (I_{N-1} - a^{\mathcal{C}} \mathcal{S}^{\mathcal{C}})^T \\ (I_{N-1} - a^{\mathcal{C}} \mathcal{S}^{\mathcal{C}}) & -I_{N-1} \end{bmatrix} < 0, \quad (32)$$

$$\mathcal{C} = 1, 2, \dots, \mathcal{K},$$

$$\begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{\alpha_2}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix} < 0, \quad (33)$$

where

$$\begin{aligned} \mathcal{S}^{\mathcal{C}} &= (\mathcal{L}_{ij}^{\mathcal{C}} - \mathcal{L}_{Nj}^{\mathcal{C}})_{(N-1) \times (N-1)}, \quad \mathcal{C} = 1, 2, \dots, \mathcal{K}, \\ \mathcal{M}_1 &= \text{diag}(\mathcal{Q}_1^+ \mathcal{Q}_1^-, \mathcal{Q}_2^+ \mathcal{Q}_2^-, \dots, \mathcal{Q}_n^+ \mathcal{Q}_n^-), \end{aligned} \quad (34)$$

$$\mathcal{M}_2 = \text{diag}\left(\frac{\mathcal{Q}_1^+ + \mathcal{Q}_1^-}{2}, \frac{\mathcal{Q}_2^+ + \mathcal{Q}_2^-}{2}, \dots, \frac{\mathcal{Q}_n^+ + \mathcal{Q}_n^-}{2}\right),$$

then, for the impulsive time sequence $\{t_h\}_{h=1}^{+\infty}$ admitting $\sup\{t_h - t_{h-1}\} \leq \alpha_1$, system (31) can achieve static multisynchronization.

Proof. From (A1)–(A3), according to Lemma 1, every subnetwork of system (1) has $(r+1)^n$ locally Mittag-Leffler stable equilibria $S_1, S_2, \dots, S_{(r+1)^n}$.

Let $\mathcal{Y}_i(t) = x_i(t) - S$, $i = 1, 2, \dots, N$, where $S \in \{S_h, \bar{h} = 1, 2, \dots, (r+1)^n\}$; from (31), then

$$\begin{aligned} {}^C D_{t_0}^q \mathcal{Y}(t) &= -(I_N \otimes D)\mathcal{Y}(t) + (I_N \otimes B)F(\mathcal{Y}(t)), \\ t &\neq t_h, \end{aligned} \quad (35)$$

$$\begin{aligned} \Delta \mathcal{Y}(t) &= -(a^{\rho(t)} L^{\rho(t)} \otimes I_n)\mathcal{Y}(t^-), \quad t = t_h, \\ t &\geq t_0 \geq 0, \quad h = 1, 2, \dots, \end{aligned}$$

where $\mathcal{Y}(t) = (\mathcal{Y}_1^T(t), \mathcal{Y}_2^T(t), \dots, \mathcal{Y}_N^T(t))^T$ and $F(\mathcal{Y}(t)) = (F^T(\mathcal{Y}_1(t)), F^T(\mathcal{Y}_2(t)), \dots, F^T(\mathcal{Y}_N(t)))^T$.

Let $\mathcal{Z}_i(t) = \mathcal{Y}_i(t) - \mathcal{Y}_N(t)$, $i = 1, 2, \dots, N-1$, and system (35) can be reformulated as

$$\begin{aligned} {}^C D_{t_0}^q \mathcal{Z}(t) &= -(I_{N-1} \otimes D)\mathcal{Z}(t) \\ &\quad + (I_{N-1} \otimes B)F(\mathcal{Z}(t)), \quad t \neq t_h, \\ {}^C D_{t_0}^q \mathcal{Y}_N(t) &= -D\mathcal{Y}_N(t) + BF(\mathcal{Y}_N(t)), \quad t \neq t_h, \\ \Delta \mathcal{Z}(t) &= -(a^{\rho(t)} \mathcal{S}^{\rho(t)} \otimes I_n)\mathcal{Z}(t^-), \quad t = t_h, \\ \Delta \mathcal{Y}_N(t) &= -(a^{\rho(t)} \widehat{\mathcal{S}}^{\rho(t)} \otimes I_n)\mathcal{Y}_N(t^-), \quad t = t_h, \\ t &\geq t_0 \geq 0, \quad h = 1, 2, \dots, \end{aligned} \quad (36)$$

where $\mathcal{Z}(t) = (\mathcal{Z}_1^T(t), \mathcal{Z}_2^T(t), \dots, \mathcal{Z}_{N-1}^T(t))^T$, $F(\mathcal{Z}(t)) = (F^T(\mathcal{Z}_1(t)), F^T(\mathcal{Z}_2(t)), \dots, F^T(\mathcal{Z}_{N-1}(t)))^T$, and $\widehat{\mathcal{S}}^{\rho(t)} = (\mathcal{L}_{N1}^{\rho(t)}, \mathcal{L}_{N2}^{\rho(t)}, \dots, \mathcal{L}_{N(N-1)}^{\rho(t)})$.

Obviously, for $i = 1, 2, \dots, N-1$,

$$\begin{aligned} \mathcal{X}_i(t) &= \mathcal{Y}_i(t) - \mathcal{Y}_N(t) = (x_i(t) - S) - (x_N(t) - S) \\ &= x_i(t) - x_N(t). \end{aligned} \quad (37)$$

According to Lemma 3, (32) is equivalent to

$$\begin{aligned} (I_{N-1} - a^{\mathcal{E}} \mathcal{S}^{\mathcal{E}})^T (I_{N-1} - a^{\mathcal{E}} \mathcal{S}^{\mathcal{E}}) &\leq \alpha_2 I_{N-1}, \\ \mathcal{E} &= 1, 2, \dots, \mathcal{K}. \end{aligned} \quad (38)$$

Define a Lyapunov function

$$V(t) = \mathcal{X}^T(t) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t). \quad (39)$$

When $t = t_h$, we can obtain

$$\begin{aligned} V(t_h) &= \mathcal{X}^T(t_h) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t_h) \\ &= \mathcal{X}^T(t_h^-) [(I_{N-1} - a^{\mathcal{E}} \mathcal{S}^{\mathcal{E}}) \otimes I_n]^T (I_{N-1} \otimes \mathcal{R}) \\ &\quad \times [(I_{N-1} - a^{\mathcal{E}} \mathcal{S}^{\mathcal{E}}) \otimes I_n] \mathcal{X}(t_h^-) \\ &\leq \alpha_2 V(t_h^-). \end{aligned} \quad (40)$$

On the other hand, by (3) in (A1), for any given $\mathcal{W} = \text{diag}(\mathcal{W}_1, \mathcal{W}_2, \dots, \mathcal{W}_n) > 0$, we have

$$\begin{aligned} 0 &\leq \sum_{j=1}^n \mathcal{W}_j [\mathcal{Q}_j^+ \mathcal{X}_j(t) - F(\mathcal{X}_j(t))] \\ &\quad \cdot [F(\mathcal{X}_j(t)) - \mathcal{Q}_j^- \mathcal{X}_j(t)] = -\mathcal{X}_i^T(t) \mathcal{W} \mathcal{M}_1 \mathcal{X}_i(t) \\ &\quad + 2\mathcal{X}_i^T(t) \mathcal{W} \mathcal{M}_2 F(\mathcal{X}_i(t)) - F^T(\mathcal{X}_i(t)) \\ &\quad \cdot \mathcal{W} F(\mathcal{X}_i(t)), \end{aligned} \quad (41)$$

and thus

$$\begin{aligned} 0 &\leq -\mathcal{X}^T(t) I_{N-1} \otimes (\mathcal{W} \mathcal{M}_1) \mathcal{X}(t) + 2\mathcal{X}^T(t) I_{N-1} \\ &\quad \otimes (\mathcal{W} \mathcal{M}_2) F(\mathcal{X}(t)) - F^T(\mathcal{X}(t)) I_{N-1} \\ &\quad \otimes \mathcal{W} F(\mathcal{X}(t)). \end{aligned} \quad (42)$$

By (33), it follows that

$$\begin{aligned} \begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{(\alpha_2 + \alpha_3)}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix} \\ < 0, \end{aligned} \quad (43)$$

where constant $\alpha_3 \in (0, 1 - \alpha_2)$.

When $t \neq t_h$, we can get

$$\begin{aligned} {}^C D_{t_0}^q V(t) &\leq 2\mathcal{X}^T(t) (I_{N-1} \otimes \mathcal{R}) \mathcal{X}(t) + 2\mathcal{X}^T(t) \\ &\quad \cdot (I_{N-1} \otimes \mathcal{R}) \\ &\quad \cdot [- (I_{N-1} \otimes D) \mathcal{X}(t) + (I_{N-1} \otimes B) F(\mathcal{X}(t))]. \end{aligned} \quad (44)$$

Together with (42)–(44),

$$\begin{aligned} {}^C D_{t_0}^q V(t) &\leq \mathcal{X}^T(t) (I_{N-1} \otimes \chi) \mathcal{X}(t) \\ &\quad - \frac{(\alpha_2 + \alpha_3)}{\alpha_1} V(t) < -\frac{(\alpha_2 + \alpha_3)}{\alpha_1} V(t), \end{aligned} \quad (45)$$

where

$$\begin{aligned} \mathcal{X}^T(t) &= (\mathcal{X}^T(t), F^T(\mathcal{X}(t)))^T, \\ \chi &= \begin{bmatrix} -\mathcal{R}D - D^T \mathcal{R} + \frac{(\alpha_2 + \alpha_3)}{\alpha_1} \mathcal{R} - \mathcal{W} \mathcal{M}_1 & (\mathcal{R}B + \mathcal{W} \mathcal{M}_2) \\ (\mathcal{R}B + \mathcal{W} \mathcal{M}_2)^T & -\mathcal{W} \end{bmatrix}. \end{aligned} \quad (46)$$

According to Lemma 4,

$$\begin{aligned} V(t) &\leq V(t_0) E_q \left(-\frac{(\alpha_2 + \alpha_3)}{\alpha_1} (t - t_0)^q \right), \\ &\quad t \geq t_0 \geq 0, \end{aligned} \quad (47)$$

so $V(t) \rightarrow 0$ as $t \rightarrow +\infty$, which implies that, for any given initial value of (1), $x_1(t) = x_2(t) = \dots = x_N(t)$ as $t \rightarrow +\infty$, and, hence, system (31) can reach complete synchronization. \square

Moreover, consider the N th subnetwork of (31):

$$\begin{aligned} {}^C D_{t_0}^q x_N(t) &= -D x_N(t) + B F(x_N(t)), \quad t \neq t_h, \\ \Delta x_N(t) &= \sum_{j=1}^{N-1} c_{Nj}^{\rho(t)} (x_j(t^-) - x_N(t^-)), \quad t = t_h, \\ &\quad t \geq t_0 \geq 0, \quad h = 1, 2, \dots \end{aligned} \quad (48)$$

Based on the above analysis, we have $\Delta x_N(t) = 0$ as $t \rightarrow +\infty$. Through Lemma 1, the N th subnetwork of (31) has $(r+1)^n$ locally Mittag-Leffler stable equilibria $S_1, S_2, \dots, S_{(r+1)^n}$. Therefore, it follows that $x_1(t) = x_2(t) = \dots = x_N(t) = S$ as $t \rightarrow +\infty$, $S \in \{S_{\tilde{h}}, \tilde{h} = 1, 2, \dots, (r+1)^n\}$. To sum up, system (31) can achieve static multisynchronization.

Remark 7. Linear matrix inequality has emerged as a very powerful tool and design technique for a lot of the control problems. From the viewpoint of mathematics, the linear matrix inequality is a convex constraint. So the computational procedure scheme for linear matrix inequality can be processed efficiently. A much more effective computing method for solving these kinds of issues is the interior point method [32]. By using Newton's method, the interior point method transforms the constrained optimization problem into an unconstrained optimization problem to be solved. Accordingly, reducing a control design problem to the linear matrix inequality may be a practical approach to this problem [33]. Given that, we have proposed the linear matrix inequality based design method for multisynchronization

control problem. From the previous discussion, linear matrix inequalities (13) and (14) in Theorem 5 and linear matrix inequalities (32) and (33) in Theorem 6 can be effectively solved.

Remark 8. When nonlinear systems generate multiple locally stable equilibria, finding appropriate control strategy and effective method to deal with such nonlinear systems is difficult. For example, as shown in [34], how to achieve the new synchronization scheme for multistable nonlinear systems is a very intractable problem. However, this issue may be solved if the effective impulsive control strategy is adopted.

Remark 9. Note that the impulsive control strategy (11) or (30) samples the state information only at impulsive times t_h ; namely, each subnetwork takes only the sampling information of its neighbors. Hence, compared with the continuous control law, the impulsive control strategy then has strong pertinence, low energy consumption, and high response speed. As revealed in (12) and (31), the impulsive control system integrates the advantages of impulsive control and continuous control.

Remark 10. Under the framework of Filippov solution, Gu et al. [1] investigate the global synchronization of fractional-order memristive neural networks based on comparison principle and Lyapunov method. Together with fractional-order differential inequality and Lyapunov theory, Xiao et al. [8] analyze the finite-time synchronization of fractional-order memristive bidirectional associative memory neural networks. By employing Holder inequality, C_p inequality, and Gronwall-Bellman inequality, Yang et al. [9] formulate the quasi-uniform synchronization for fractional-order memristive neural networks. Based on Barbalat lemma and Razumikhin-type stability theorem, Zhang et al. [11] establish projective synchronization for fractional-order memristive neural networks. By using the infinitesimal generator on analytic semigroup principle and inequality techniques, Zhou et al. [12] study exponential synchronization of stochastic neural networks driven by fractional Brownian motion. By introducing the concept of joint connectivity and sequential connectivity, Chen et al. [21] show that complex networks can synchronize even if the topology is not connected at any time instant. By combining adaptive control and impulsive control, Yang et al. [17] discuss the global exponential synchronization of complex dynamical networks with nonidentical nodes and stochastic perturbation. By using the mathematical induction method, Zhang et al. [18] achieve the stochastic exponential synchronization for a class of delayed dynamical networks under delayed impulsive control, whereas the above works are all concerned about the global synchronization (or monosynchronization). These analytical methods for global synchronization (or monosynchronization) can not be migrated well to the multisynchronization problem. Using the impulsive control strategy and the Razumikhin-type technique, Wang et al. [16] study the multisynchronization problem of coupled neural networks with directed topology. Nevertheless, the controlled system in [16] is integer-order model. As have often been noted in most existing publications, analytical approach for integral-order systems could not be directly

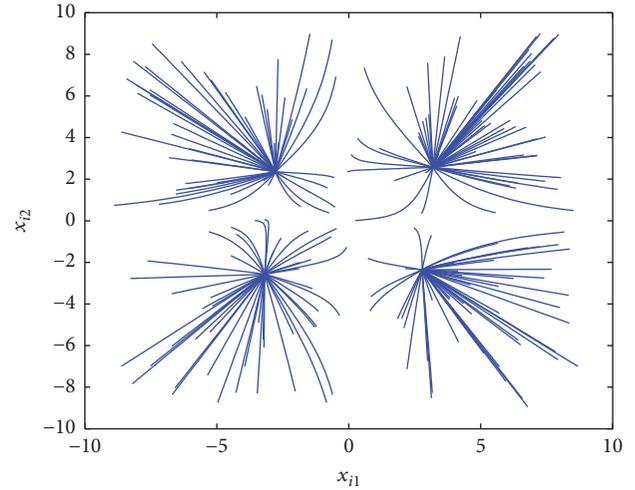


FIGURE 1: State evolutive characteristics.

extended and applied to deal with fractional-order systems. In this light, this paper extends and renews the relative results.

4. Two Numerical Examples

In the following numerical examples, we consider the coupled fractional-order neural networks consisting of three identical subnetworks; meanwhile, every subnetwork has two neuron states and the parameters of every subnetwork are characterized by

$$D = \text{diag}(1, 1),$$

$$B = \begin{bmatrix} 3 & 0.2 \\ 0.1 & 2.5 \end{bmatrix}, \quad (49)$$

$$f_1(s) = f_2(s) = \tanh(s).$$

When there is zero input in every subnetwork above, conditions (2)–(6) are satisfied. That is, (A1)–(A3) hold. According to Lemma 1, every subnetwork has 4 locally Mittag-Leffler stable equilibria. Figure 1 depicts the state evolutive characteristics.

Example 11. We investigate fixed topology case. The Laplacian matrix $L = (\mathcal{L}_{ij})_{3 \times 3}$ associated with digraph \mathcal{D} is described as

$$L = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix}. \quad (50)$$

To choose constant $\alpha_1 = 0.3$, in addition, we select $\alpha_2 = 0.9$, $a = 0.1$, and matrices $\mathcal{R} = \text{diag}(1, 1)$ and $\mathcal{W} = \text{diag}(0.5, 0.5)$, and then conditions (13) and (14) are satisfied. That is, Theorem 5 holds. According to Theorem 5, the controlled system can achieve static multisynchronization. Figure 2 depicts the multisynchronization characteristics.

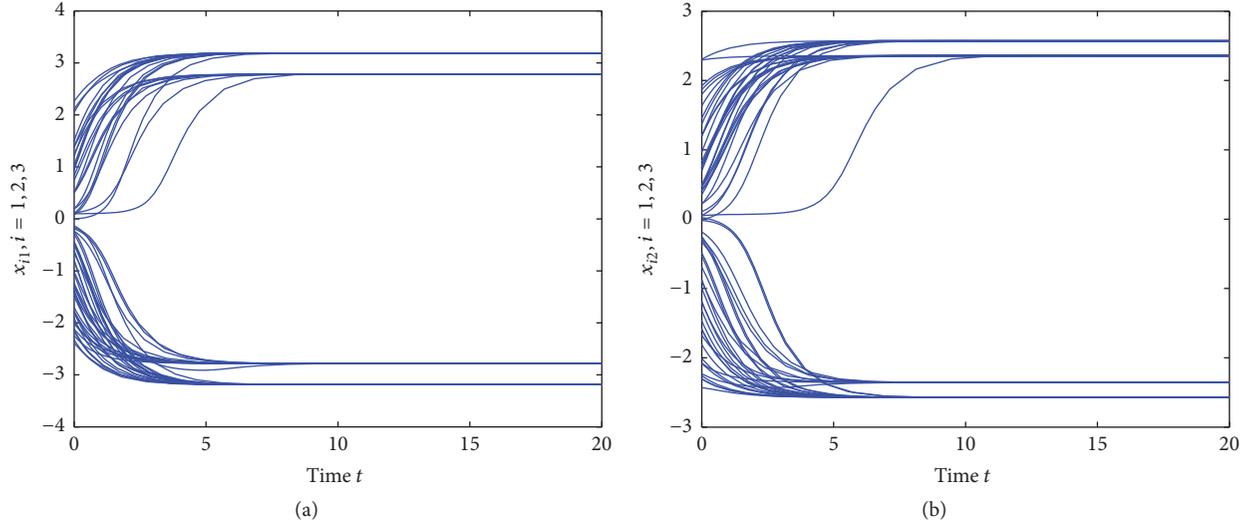


FIGURE 2: Multisynchronization characteristics in fixed topology case.

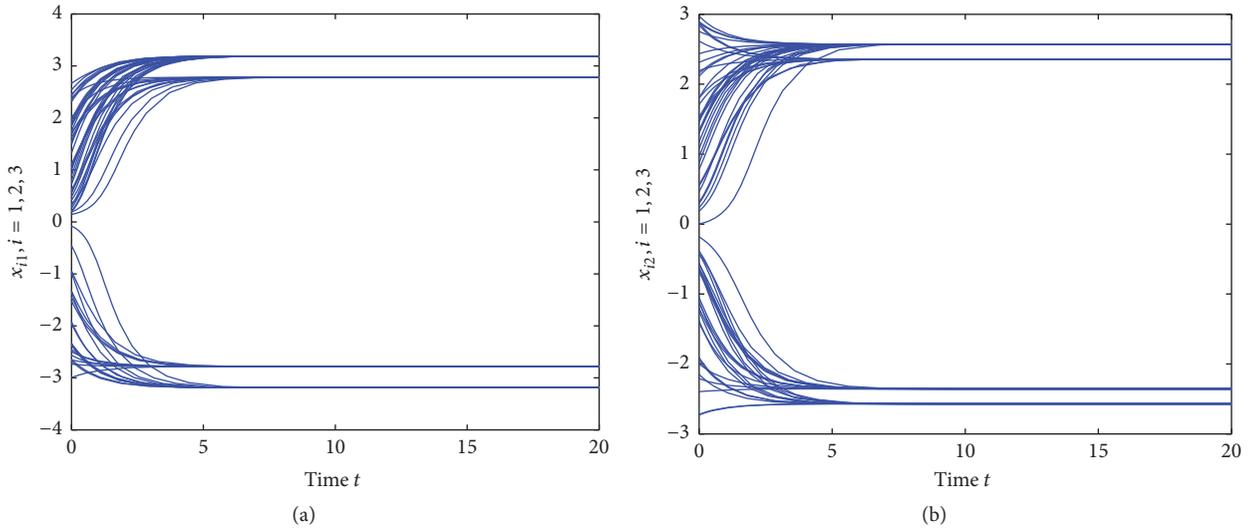


FIGURE 3: Multisynchronization characteristics in switching topology case.

Example 12. We investigate switching topology case. The Laplacian matrices $L^{\rho(t)} = (\mathcal{L}_{ij}^{\rho(t)})_{3 \times 3}$ associated with digraphs $\mathcal{D}^{\rho(t)}$ are described as

$$L^1 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix},$$

$$L^2 = \begin{bmatrix} 1 & 0 & -1 \\ -1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix},$$

$$L^3 = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}.$$

(51)

To choose constant $\alpha_1 = 0.3$, in addition, we select $\alpha_2 = 0.9$, $a^1 = 0.1$, $a^2 = 0.1$, $a^3 = 0.3$, and matrices $\mathcal{R} = \text{diag}(1, 1)$ and $\mathcal{W} = \text{diag}(0.5, 0.5)$, and then conditions (32) and (33) are satisfied. That is, Theorem 6 holds. According to Theorem 6, the controlled system can achieve static multisynchronization. Figure 3 depicts the multisynchronization characteristics.

5. Concluding Remarks

Multisynchronization represents one of the most striking manifestations of multiconsistency in multistable systems. In this paper, we present the analytical results on the multisynchronization for coupled multistable fractional-order neural networks. Using the impulsive control principle, fractional-order Lyapunov method, and linear matrix inequality technique, several sufficient conditions are deduced to guarantee multisynchronization which characterize the cardinality of the set of synchronization manifolds. How to develop new multisynchronization schemes for multistable fractional-order systems would be the topic of future research.

Conflicts of Interest

The author declares that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The work is supported by the Research Project of Hubei Provincial Department of Education of China under Grant T201412.

References

- [1] Y. Gu, Y. Yu, and H. Wang, "Synchronization for fractional-order time-delayed memristor-based neural networks with parameter uncertainty," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 353, no. 15, pp. 3657–3684, 2016.
- [2] P. Liu, Z. Zeng, and J. Wang, "Multiple Mittag-Leffler Stability of Fractional-Order Recurrent Neural Networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2279–2288, 2017.
- [3] R. Rakkiyappan, J. Cao, and G. Velmurugan, "Existence and uniform stability analysis of fractional-order complex-valued neural networks with time delays," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 1, pp. 84–97, 2015.
- [4] G. Velmurugan, R. Rakkiyappan, V. Vembarasan, J. Cao, and A. Alsaedi, "Dissipativity and stability analysis of fractional-order complex-valued neural networks with time delay," *Neural Networks*, vol. 86, pp. 42–53, 2017.
- [5] A. Wu and Z. Zeng, "Boundedness, Mittag-Leffler stability and asymptotical ω -periodicity of fractional-order fuzzy neural networks," *Neural Networks*, vol. 74, pp. 73–84, 2016.
- [6] A. Wu and Z. Zeng, "Global Mittag-Leffler stabilization of fractional-order memristive neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 1, pp. 206–217, 2017.
- [7] A. Wu, L. Liu, T. Huang, and Z. Zeng, "Mittag-Leffler stability of fractional-order neural networks in the presence of generalized piecewise constant arguments," *Neural Networks*, vol. 85, pp. 118–127, 2017.
- [8] J. Xiao, S. Zhong, Y. Li, and F. Xu, "Finite-time Mittag-Leffler synchronization of fractional-order memristive BAM neural networks with time delays," *Neurocomputing*, vol. 219, pp. 431–439, 2017.
- [9] X. Yang, C. Li, T. Huang, Q. Song, and X. Chen, "Quasi-uniform synchronization of fractional-order memristor-based neural networks with delay," *Neurocomputing*, vol. 234, pp. 205–215, 2017.
- [10] L. Zhang, Q. Song, and Z. Zhao, "Stability analysis of fractional-order complex-valued neural networks with both leakage and discrete delays," *Applied Mathematics and Computation*, vol. 298, pp. 296–309, 2017.
- [11] L. Zhang, Y. Yang, and F. Wang, "Projective synchronization of fractional-order memristive neural networks with switching jumps mismatch," *Physica A. Statistical Mechanics and its Applications*, vol. 471, pp. 402–415, 2017.
- [12] W. Zhou, X. Zhou, J. Yang, Y. Liu, X. Zhang, and X. Ding, "Exponential synchronization for stochastic neural networks driven by fractional Brownian motion," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 353, no. 8, pp. 1689–1712, 2016.
- [13] C. D. Cruz-Ancona, R. Martínez-Guerra, and C. A. Pérez-Pinacho, "Generalized multi-synchronization: a leader-following consensus problem of multi-agent systems," *Neurocomputing*, vol. 233, pp. 52–60, 2017.
- [14] R. Martinez-Guerra, C. D. Cruz-Ancona, and C. A. Perez-Pinacho, "Generalized multi-synchronization viewed as a multi-agent leader-following consensus problem," *Applied Mathematics and Computation*, vol. 282, pp. 226–236, 2016.
- [15] H. Salarieh and M. Shahrokhi, "Multi-synchronization of chaos via linear output feedback strategy," *Journal of Computational and Applied Mathematics*, vol. 223, no. 2, pp. 842–852, 2009.
- [16] Y.-W. Wang, W. Yang, J.-W. Xiao, and Z.-G. Zeng, "Impulsive multisynchronization of coupled multistable neural networks with time-varying delay," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1560–1571, 2017.
- [17] X. Yang, J. Cao, and J. Lu, "Stochastic synchronization of complex networks with nonidentical nodes via hybrid adaptive and impulsive control," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 2, pp. 371–384, 2012.
- [18] W. Zhang, Y. Tang, Q. Miao, and J.-A. Fang, "Synchronization of stochastic dynamical networks under impulsive control with time delays," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 10, pp. 1758–1768, 2014.
- [19] M. Ayati and H. Khaloozadeh, "Designing a novel adaptive impulsive observer for nonlinear continuous systems using LMIs," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 59, no. 1, pp. 179–187, 2012.
- [20] W.-H. Chen, D. Wei, and W. X. Zheng, "Delayed impulsive control of takagi-sugeno fuzzy delay systems," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 3, pp. 516–526, 2013.
- [21] Y. Chen, W. Yu, F. Li, and S. Feng, "Synchronization of complex networks with impulsive control and disconnected topology," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 60, no. 5, pp. 292–296, 2013.
- [22] M. Claeys, D. Arzelier, D. Henrion, and J.-B. Lasserre, "Measures and LMIs for impulsive nonlinear optimal control," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1374–1379, 2014.
- [23] L. Ding, P. Yu, Z.-W. Liu, and Z.-H. Guan, "Consensus and performance optimisation of multi-agent systems with position-only information via impulsive control," *IET Control Theory & Applications*, vol. 7, no. 1, pp. 16–24, 2013.
- [24] W. Du, S. Y. S. Leung, Y. Tang, and A. V. Vasilakos, "Differential evolution with event-triggered impulsive control," *IEEE Transactions on Cybernetics*, vol. 47, no. 1, pp. 244–257, 2017.

- [25] S. L. Fraga and F. L. Pereira, "Hamilton-Jacobi-Bellman equation and feedback synthesis for impulsive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 1, pp. 244–249, 2012.
- [26] D. He and L. Xu, "Ultimate Boundedness of Nonautonomous Dynamical Complex Networks under Impulsive Control," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 10, pp. 997–1001, 2015.
- [27] X. Li and S. Song, "Stabilization of Delay Systems: delay-dependent Impulsive Control," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 406–411, 2017.
- [28] Y. Li, Y. Sun, J. Hua, and L. Li, "Indirect adaptive type-2 fuzzy impulsive control of nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 23, no. 4, pp. 1084–1099, 2015.
- [29] X. Liu and K. Zhang, "Impulsive control for stabilisation of discrete delay systems and synchronisation of discrete delay dynamical networks," *IET Control Theory & Applications*, vol. 8, no. 13, pp. 1185–1195, 2014.
- [30] Y. Liu, S. Zhao, and J. Lu, "A new fuzzy impulsive control of chaotic systems based on T-S fuzzy model," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 2, pp. 393–398, 2011.
- [31] X. Lu, N. Chen, Y. Wang, L. Qu, and J. Lai, "Distributed impulsive control for islanded microgrids with variable communication delays," *IET Control Theory & Applications*, vol. 10, no. 14, pp. 1732–1739, 2016.
- [32] T. H. Cormen, C. E. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, The MIT Press, Cambridge, Massachusetts, Mass, USA, 2009.
- [33] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, SIAM, Philadelphia, PA, USA, 1994.
- [34] A. Wu and Z. Zeng, "Output convergence of fuzzy neurodynamic system with piecewise constant argument of generalized type and time-varying input," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 12, pp. 1689–1702, 2016.

Research Article

Adaptive Neural Output Feedback Control for Uncertain Robot Manipulators with Input Saturation

Rong Mei and ChengJiang Yu

Criminal Investigation Department, Nanjing Forest Police College, Nanjing 210023, China

Correspondence should be addressed to Rong Mei; meirongnuaa@163.com

Received 8 May 2017; Accepted 4 July 2017; Published 7 August 2017

Academic Editor: Yanan Li

Copyright © 2017 Rong Mei and ChengJiang Yu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents an adaptive neural output feedback control scheme for uncertain robot manipulators with input saturation using the radial basis function neural network (RBFNN) and disturbance observer. First, the RBFNN is used to approximate the system uncertainty, and the unknown approximation error of the RBFNN and the time-varying unknown external disturbance of robot manipulators are integrated as a compounded disturbance. Then, the state observer and the disturbance observer are proposed to estimate the unmeasured system state and the unknown compounded disturbance based on RBFNN. At the same time, the adaptation technique is employed to tackle the control input saturation problem. Utilizing the estimate outputs of the RBFNN, the state observer, and the disturbance observer, the adaptive neural output feedback control scheme is developed for robot manipulators using the backstepping technique. The convergence of all closed-loop signals is rigorously proved via Lyapunov analysis and the asymptotically convergent tracking error is obtained under the integrated effect of the system uncertainty, the unmeasured system state, the unknown external disturbance, and the input saturation. Finally, numerical simulation results are presented to illustrate the effectiveness of the proposed adaptive neural output feedback control scheme for uncertain robot manipulators.

1. Introduction

In the past decades, there have been enormous research efforts in the development of efficient control schemes for robot manipulators [1–3]. In general, robot manipulators have nonlinear characteristics and are subject to parameter perturbations and unknown disturbance [4, 5]. Thus, the model is dynamically complex when the robot manipulator is operated in some uncertain surroundings such as underwater or in space. To achieve satisfactory control performance, a number of robust control schemes have been developed for uncertain robot manipulators, among them are universal approximator based control [6], disturbance observer based control [7], and sliding mode control [8]. To successfully complete the required tasks, it is necessary to precisely control the uncertain and nonlinear robot manipulators [9]. In [10], the trajectory and force tracking control was studied for constrained mobile manipulators with parameter uncertainty. In [11], a robust tracking control scheme was proposed for rigid robotic manipulators with uncertain dynamics. In [12],

an adaptive neural network tracking control was proposed for manipulators with uncertain kinematics, dynamics, and actuator model. However, robot manipulators in operations are frequently subjected to the integrated effect of the system uncertainty, the time-varying unknown external disturbance, and the input saturation constraint. Thus, the robust tracking control schemes should be further developed for the robot manipulator to manage uncertainties, disturbances, nonlinearities, input constraints, and their coupling effects.

When dealing with system uncertainties, neural networks (NNs), being a universal approximator with learning ability, can be employed to compensate for the uncertain nonlinear dynamics in the robot manipulator [13–15]. In [16], an adaptive control was studied for robot manipulators with NN based compensation of frictional uncertainties. Tracking control was developed based on NN strategy for the robot manipulator [17]. In [18], the robust adaptive control was studied for robot manipulators using generalized fuzzy neural networks. Robust neural-fuzzy-network control was developed for the robot manipulator including actuator dynamics

in [19]. Generally speaking, the joint position measurements can be obtained by means of encoders, which can give very accurate measurements of joint displacements. On the contrary, joint velocity measurements are more problematic due to the noise contamination nature; one solution is to use a state observer. With the state observer incorporated, the output feedback tracking controllers can be developed to make the whole closed-loop control with only position measurements [20]. To handle the unmeasured states of nonlinear systems, adaptive output feedback control methods were proposed for various nonlinear systems in [21–26]. In [27], the adaptive control law was studied for robot manipulators without velocity feedback. Neural network output feedback control was proposed for robot manipulators in [28]. In [29], the output feedback control was developed for robot manipulators based on deterministic learning method. An adaptive output feedback controller was studied for robot arms in [30]. However, the time-varying unknown disturbance and the approximation error of neural networks are usually unknown which need to be efficiently tackled in the output feedback control design.

Because robotic manipulators are subject to different types of disturbances in the changeable work environment, the control performance will be degraded. Thus, it is necessary to develop the robust control scheme to handle the unknown disturbance [31]. To fully utilize the dynamic information of the unknown external disturbance, the disturbance observer can be employed to design the robust output control scheme for the robot manipulator [32–34]. Recently, many disturbance observers have been designed to compensate for the disturbance effect. In [35], the disturbance observer based control (DOBC) was proposed for complex continuous models. The disturbance attenuation and rejection problems were investigated for a class of multi-input and multioutput (MIMO) nonlinear systems in [36]. Sliding mode control was presented for systems with mismatched uncertainties based on disturbance observer in [37]. Due to the significance of attenuating the effect of the unknown disturbance, the disturbance observer has been extensively used in robot manipulator systems [38]. The nonlinear disturbance observer was developed for robot manipulators in [39]. In [40], decentralized adaptive robust control was proposed for robot manipulators using disturbance observers. Nonlinear disturbance observer design was presented for robotic manipulators in [41]. In [42], Lyapunov-based nonlinear disturbance observer was developed for serial n -link robot manipulators. Robust constrained control was proposed for MIMO nonlinear systems based on disturbance observer in [43]. Although the disturbance observer technique has some practical applications in robotic manipulator control area, the disturbance observer based output feedback control strategy needs to be further investigated to enhance the antidisturbance ability by combining with the NN.

When parameter change, disturbances, or state errors become large, the required input torques will quickly reach saturation due to the needed massive control effort to maintain tracking control performance for robotic manipulators [44–52]. Under this case, the unchanged control output will destroy the stability of the closed-loop control systems.

Thus, in trajectory control of robot manipulators, the input saturation should be explicitly considered to eliminate the saturation effect [53]. The control of robot manipulators with bounded input was studied in [54]. In [55], the robust control was developed for robot manipulators with torque saturation using fuzzy logic. Fuzzy saturated output feedback tracking control was proposed for robot manipulators using a singular perturbation theory based approach in [56]. In [57], a saturated output feedback tracking control was studied for robot manipulators via fuzzy self-tuning. Disturbance observer based path tracking control was developed for robot manipulator considering torque saturation in [58]. Robust adaptive motion/force tracking control was proposed for uncertain constrained robot manipulators in [59]. In [60], a hybrid fuzzy adaptive output feedback control design was proposed for uncertain MIMO nonlinear systems with time-varying delays and input saturation. In this paper, the adaptive neural output feedback control scheme will be developed by using the RBFNN and disturbance observer for uncertain robot manipulators with input saturation based on backstepping technique. Backstepping control as an efficient control method of nonlinear systems can provide a systematic framework for the tracking control of robot manipulators. Its design flexibility has led to the robust adaptive backstepping control being extensively studied in the nonlinear control system design including the robust control design of robot manipulators [61, 62].

This work is motivated by the disturbance observer-based adaptive neural output feedback control to follow desired time-varying trajectories of robot manipulators. The state observer is designed to estimate the unmeasured state, and the disturbance observer and the RBFNN are employed to suppress the effect of the system uncertainty and the unknown external disturbance. The organization of the paper is as follows. Section 2 presents the problem descriptions of the adaptive neural output feedback control for the robot manipulator. Section 3 describes the adaptive neural output feedback control design using the state observer, the disturbance observer, and the RBFNN based on backstepping method. Simulation studies are provided in Section 4 to demonstrate the effectiveness of the proposed disturbance observer based adaptive neural output feedback control approach, followed by some concluding remarks in Section 5.

2. Problem Formulation

The dynamics of an uncertain n -joint robot manipulator can be described as [63]

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) = u(t), \quad (1)$$

where $q(t)$ is the $R^{n \times 1}$ state vector of joint angular positions; $u(t)$ is the $R^{n \times 1}$ control input vector of applied joint torques; $M(q)$ is the $R^{n \times n}$ symmetric positive-definite inertia matrix which is unknown; $C(q, \dot{q})$ is the unknown $R^{n \times n}$ matrix of Coriolis and centrifugal torques; and $g(q)$ is the unknown $R^{n \times 1}$ vector of gravitational torques.

Defining $x = (q^T, \dot{q}^T)^T$ and considering (7), the uncertain robot manipulator dynamics (1) can be written as

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= F(x) + G_0(x_1)u(t),\end{aligned}\quad (2)$$

where $x_1 = q$, $x_2 = \dot{q}$, $F(x) = M(q)^{-1}(-C(q, \dot{q})\dot{q} - g(q))$, and $G_0(x_1) = M(q)^{-1}$. Since $C(q, \dot{q})$ and $g(q)$ are unknown, $F(x)$ is also unknown which denotes the system uncertainty.

In general, the robot manipulator may suffer from the unknown external disturbance $d(t) \in R^{n \times 1}$. Here, the uncertain robot manipulator dynamics (2) can be expressed as

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= F(x) + G_0(x_1)u(t) + d(t).\end{aligned}\quad (3)$$

In most practical operations, there usually exist the control input saturation $u = \text{sat}(v)$, where v is a desired control command. Thus, the uncertain robot manipulator dynamics (3) with input saturation can be written as

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= F(x) + G_0(x_1)\text{sat}(v) + d(t).\end{aligned}\quad (4)$$

The function $\text{sat}(\cdot)$ represents the input saturation of robot manipulator systems and is defined as

$$\text{sat}(v_i) = \text{sign}(u_i) \min\{u_{\max i}, |u_i|\}, \quad i = 1, 2, \dots, m, \quad (5)$$

where $u_{\max i}$ represents the saturation level of the i th input and is assumed to be known.

On the other hand, we assume the symmetric positive-definite inertia matrix $M(q)$ with uncertainty $\Delta M(q)$. Thus, we have

$$G_0(x_1) = G(x_1) + \Delta G(x_1), \quad (6)$$

where $G(x_1)$ is the known nominal control gain matrix and $\Delta G(x_1)$ is the uncertainty of the control gain matrix.

Substituting (6) into (4) yields

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= F(x) + G(x_1)\text{sat}(v) + \Delta G(x_1)\text{sat}(v) + d(t).\end{aligned}\quad (7)$$

To handle the uncertainty $F(x)$ of the robot manipulator, the following lemma is used.

Lemma 1 (see [64]). *As a class of linearly parameterized NN, RBFNNs are adopted to approximate the continuous function $f(Z) : R^q \rightarrow R$ and can be expressed as follows:*

$$f(Z) = \widehat{\Theta}^T \phi(Z) + \varepsilon, \quad (8)$$

where $Z = [z_1, z_2, \dots, z_q]^T \in R^q$ is the input vector of the NN, $\widehat{\Theta} \in R^p$ is a weight vector of the NN, $\phi(Z) = [\phi_1(Z), \phi_2(Z), \dots, \phi_p(Z)]^T \in R^p$ is the basis function, and ε is the approximation

error of the NN. The optimal weight value Θ^* of RBFNN is given by

$$\Theta^* = \arg \min_{\Theta \in \Omega_f} \left[\sup_{z \in S_Z} |\widehat{f}(Z | \widehat{\Theta}) - f(Z)| \right], \quad (9)$$

where $\Omega_f = \{\widehat{\Theta} : \|\widehat{\Theta}\| \leq \overline{M}\}$ is a valid field of the estimate parameter $\widehat{\Theta}$, \overline{M} is a design parameter, and $S_Z \subset R^n$ is an allowable set of the state vector. Using the optimal weight value yields

$$\begin{aligned}f(Z) &= \Theta^{*T} \phi(Z) + \varepsilon^* \\ |\varepsilon^*| &\leq \bar{\varepsilon},\end{aligned}\quad (10)$$

where ε^* is the optimal approximation error and $\bar{\varepsilon} > 0$ is the upper bound of the approximation error.

The RBFNN is employed to approximate the system uncertainty $F(x)$ in (4) which can be expressed as $F(x) = \Theta^{*T} \phi(Z) + \varepsilon$ according to (8). Considering (4), we have

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= G(x_1)\text{sat}(v) + \Theta^{*T} \phi(Z) + \varepsilon^* + \Delta G(x_1)\text{sat}(v) \\ &\quad + d(t),\end{aligned}\quad (11)$$

where $Z = [x_1, x_2]^T$.

Due to Z including x_2 , it cannot directly be used in the RBFNN approximation. To solve this problem, by adding a term and subtracting a term, (11) can be rewritten as

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= G(x_1)\text{sat}(v) + \Theta^{*T} \phi(\widehat{Z}) + \Theta^{*T} \phi(Z) \\ &\quad - \Theta^{*T} \phi(\widehat{Z}) + \varepsilon^* + \Delta G(x_1)\text{sat}(v) + d(t),\end{aligned}\quad (12)$$

where $\widehat{Z} = [\widehat{x}_1, \widehat{x}_2]^T$ and \widehat{x}_1 and \widehat{x}_2 are the estimates of x_1 and x_2 , respectively.

Defining $D(t) = \Theta^{*T} \phi(Z) - \Theta^{*T} \phi(\widehat{Z}) + \varepsilon^* + \Delta G(x_1)\text{sat}(v) + d(t)$ and the system output choosing x_1 , we obtain

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= G(x_1)\text{sat}(v) + \Theta^{*T} \phi(\widehat{Z}) + D(x, t) \\ y &= x_1,\end{aligned}\quad (13)$$

where $D(t) \in R^{n \times 1}$ is called the system compounded disturbance which is unknown and $y \in R^{n \times 1}$ is the system output.

For the continuous desired joint trajectory $x_{1d} = y_d$, the control objective is that the adaptive neural output feedback control input v is designed to make the tracking error of each joint asymptotically convergent in the presence of system uncertainties, time-varying unknown external disturbances,

and the input saturation. To further proceed with the adaptive neural output feedback control design, the following assumptions and lemma are needed.

Assumption 2 (see [65]). The nominal inertia matrix $M_0(q)$ of the robot manipulator is uniformly bounded; that is, $a_m I \leq M_0(q) \leq a_M I$ with $a_m > 0$ and $a_M > 0$, $\forall q \in R^n$.

Assumption 3. There exist unknown positive constants θ_0 and θ_1 such that the system compounded disturbance D and its time derivative \dot{D} are bounded; that is, $\|D\| \leq \theta_0$ and $\|\dot{D}\| \leq \theta_1$.

Assumption 4. To the desired joint trajectory y_d , there exists an unknown positive constant ϵ_i such that $\|y_d^{(i)}\| \leq \epsilon_i$, $i = 1, 2$.

Assumption 5. For an uncertain n -joint robot manipulator (1) subject to the input saturation (5) and the desired reference signal y_d , there exists a feasible actual control input v which can achieve the given tracking control objective. For the difference $\Delta u = \text{sat}(v) - v$, without loss of generality, we assume that there exists an unknown constant $\xi \in R^{n \times 1}$ to render $\|\Delta u\| \leq \xi$.

Lemma 6 (see [66, 67]). *For bounded initial conditions, if there exists a C^1 continuous and positive-definite Lyapunov function $V(x)$ satisfying $\gamma_1(\|x\|) \leq V(x) \leq \gamma_2(\|x\|)$, such that $\dot{V}(x) \leq -\kappa V(x) + c$, where $\gamma_1, \gamma_2 : R^n \rightarrow R$ are class K functions and κ and c are positive constants, then the solution $x(t)$ is uniformly bounded.*

Remark 7. For a practical robot manipulator, the input saturation should meet the physical requirement of system control. In other words, there should exist an output feedback control based on the state observer and disturbance observer that can track the given desired output of the robot manipulator in the presence of the unknown external disturbance and the input saturation for all given initial conditions. Namely, the controllability of the studied robot manipulator (1) should be satisfied under the control input saturation. To guarantee the controllability of the studied robot manipulator (1), the difference $\Delta u = \text{sat}(v) - v$ between $\text{sat}(v)$ and v should be bounded from a practical view. Otherwise, the controllability of the studied robot manipulator (1) cannot be satisfied. To design the adaptive output feedback control scheme, we assume that there exists an unknown constant ξ to render $\|\Delta u\| \leq \rho(x)\xi$. Thus, Assumption 5 is reasonable.

Remark 8. To develop the disturbance observer that estimates the unknown system compounded disturbance, the bounded assumptions for the system compounded disturbance D and its time derivative \dot{D} are required. For the robot manipulator, the approximation error of the RBFNN and the external disturbance are always bounded. In addition, Θ^* , $\phi(Z)$, and $\phi(\widehat{Z})$ are also bounded. Thus, D is bounded. At the same time, from [28], we know that the approximation error $\bar{\omega} = \Theta^{*T}\phi(Z) - \Theta^{*T}\phi(\widehat{Z})$ is bounded. Thus, the derivatives of $\bar{\omega}$, ε , and d are also bounded. From above

analysis, Assumption 3 is reasonable due to the unknown constants θ_0 and θ_1 .

3. Adaptive Output Feedback Control Using Disturbance Observer

In a practical robot manipulator, the angular velocity signal is not easily obtained and is often assumed unavailable. For the unmeasured system state x and the unknown compounded disturbance $D(x, t)$ shown in (13), they cannot be directly used to design the robust adaptive control scheme for the robot manipulator based on backstepping technique. To fully utilize the dynamic information of the compounded $D(x, t)$, the disturbance observer will be proposed to estimate it. Using outputs of the RBFNN, the state observer, and the disturbance observer, the adaptive output feedback control will be designed in this section.

3.1. Design of State Observer and Disturbance Observer. To estimate the unmeasured system state, the state observer is designed as follows:

$$\begin{aligned} \hat{x}_1 &= z_1 \\ \hat{x}_2 &= z_2 + L_2(x_1 - \hat{x}_1), \end{aligned} \quad (14)$$

where the intermediate variables z_1 and z_2 are given by

$$\begin{aligned} \dot{z}_1 &= z_2 + L_1(x_1 - \hat{x}_1) \\ \dot{z}_2 &= \widehat{\Theta}^T \phi(\widehat{Z}) + \widehat{D} + G(x_1) \text{sat}(v) + L_2(x_1 - \hat{x}_1), \end{aligned} \quad (15)$$

where $L_1 = L_1^T > 0$ and $L_2 = L_2^T > 0$ are the design parameter matrices of state observer to be determined, $\widehat{\Theta}$ is the estimate of neural network value Θ^* , and \widehat{D} is the disturbance estimate of unknown compounded disturbance D .

Differentiating (14) and considering (15) yield

$$\begin{aligned} \dot{\hat{x}}_1 &= \hat{x}_2 + (L_1 - L_2)(x_1 - \hat{x}_1) \\ \dot{\hat{x}}_2 &= \widehat{\Theta}^T \phi(\widehat{Z}) + G(x_1) \text{sat}(v) + \widehat{D} + L_2(x_1 - \hat{x}_1) \\ &\quad + L_2(\dot{x}_1 - \dot{\hat{x}}_1). \end{aligned} \quad (16)$$

Invoking (13) and (16), we have

$$\begin{aligned} L_2(\dot{x}_1 - \dot{\hat{x}}_1) &= L_2(x_2 - z_2 - L_1(x_1 - \hat{x}_1)) \\ &= L_2(x_2 - \hat{x}_2 - (L_1 - L_2)(x_1 - \hat{x}_1)). \end{aligned} \quad (17)$$

Define $\bar{x}_1 = \hat{x}_1 - x_1$ and $\bar{x}_2 = \hat{x}_2 - x_2$. Then, considering (13), (16), and (17), we obtain

$$\begin{aligned} \dot{\bar{x}}_1 &= \bar{x}_2 - (L_1 - L_2)\bar{x}_1 \\ \dot{\bar{x}}_2 &= \widehat{\Theta}^T \phi(\widehat{Z}) - \Theta^{*T} \phi(\widehat{Z}) + \widehat{D} - D - L_2\bar{x}_1 - L_2\bar{x}_2 \\ &\quad + L_2(L_1 - L_2)\bar{x}_1 \\ &= -\widetilde{\Theta}^T \phi(\widehat{Z}) + \widetilde{D} - L_2\bar{x}_2 \\ &\quad + (L_2(L_1 - L_2) - L_2)\bar{x}_1, \end{aligned} \quad (18)$$

where $\widetilde{\Theta} = \Theta^* - \widehat{\Theta}$ and $\widetilde{D} = \widehat{D} - D$.

The state estimate error system (18) can be written as

$$\begin{aligned}\dot{\tilde{x}} &= A\tilde{x} + B(-\bar{\Theta}^T\phi(\bar{Z}) + \bar{D}) \\ \hat{y} &= \hat{x}_1,\end{aligned}\quad (19)$$

where $\tilde{x} = [\tilde{x}_1, \tilde{x}_2]^T$, $A = \begin{bmatrix} -(L_1-L_2) & I_{n \times n} \\ L_2(L_1-L_2)-L_2 & -L_2 \end{bmatrix}$, and the coefficients L_i , $i = 1, 2$ are design matrix parameters and $B = [0_{n \times n}, I_{n \times n}]^T$.

To estimate the unknown compounded disturbance $D(x, t)$ in (13), the disturbance observer is proposed as

$$\dot{\hat{D}} = -\hat{D} + L_3(x_1 - \hat{x}_1), \quad (20)$$

where $L_3 = L_3^T > 0$ is a design matrix parameter.

Considering $\bar{D} = \hat{D} - D$, $\tilde{x}_1 = \hat{x}_1 - x_1$, and (20) yields

$$\begin{aligned}\dot{\bar{D}} &= -\bar{D} + L_3(x_1 - \hat{x}_1) - \dot{D} - D \\ &= -\bar{D} - L_3\tilde{x}_1 - \dot{D} - D.\end{aligned}\quad (21)$$

Invoking the state estimate error system (19) and the disturbance estimate error system (21), we have

$$\begin{aligned}\dot{\tilde{X}} &= \bar{A}\tilde{X} + \bar{B}\eta \\ \hat{y} &= \hat{x}_1,\end{aligned}\quad (22)$$

where $\tilde{X} = [\tilde{x}_1, \tilde{x}_2, \bar{D}]^T$, $\bar{A} = \begin{bmatrix} -(L_1-L_2) & I_{n \times n} & 0_{n \times n} \\ L_2(L_1-L_2)-L_2 & -L_2 & I_{n \times n} \\ -L_3 & 0_{n \times n} & -I_{n \times n} \end{bmatrix}$, and the coefficients L_i , $i = 1, 2, 3$, should be chosen to render that the matrix \bar{A} is stable, $\bar{B} = \begin{bmatrix} 0_{n \times n} & 0_{n \times n} & 0_{n \times n} \\ 0_{n \times n} & I_{n \times n} & 0_{n \times n} \\ 0_{n \times n} & 0_{n \times n} & I_{n \times n} \end{bmatrix}^T$, and $\eta = [0, -\bar{\Theta}^T\phi(\bar{Z}), -\dot{D} - D]^T$.

For the designed matrix \bar{A} , given a matrix $P = P^T > 0$, there exists a positive-definite matrix $Q = Q^T > 0$ such that the following matrix inequality always hold:

$$\bar{A}^T P + P\bar{A} \leq -Q. \quad (23)$$

Consider the Lyapunov candidate as

$$V_0 = \tilde{X}^T P \tilde{X}. \quad (24)$$

The time derivative of V_0 is

$$\begin{aligned}\dot{V}_0 &= \dot{\tilde{X}}^T P \tilde{X} + \tilde{X}^T P \dot{\tilde{X}} \\ &= \tilde{X}^T (\bar{A}^T P + P\bar{A}) \tilde{X} + 2\tilde{X}^T P \bar{B} \eta \\ &\leq -\tilde{X}^T Q \tilde{X} + 2\tilde{X}^T P \bar{B} \eta.\end{aligned}\quad (25)$$

Define P_1 and \bar{B}_1 as the corresponding matrix blocks of P and \bar{B} for the state estimate error \tilde{x} and P_2 and \bar{B}_2 as the

corresponding matrix blocks of P and \bar{B} for the disturbance estimate error \bar{D} . Considering the following facts

$$\begin{aligned}-2\tilde{x}^T P_1 \bar{B}_1 \bar{\Theta}^T \phi(\bar{Z}) &\leq L_4 \|\tilde{X}\|^2 + \frac{1}{\lambda_1} \|\bar{\Theta}\|^2 \\ -2\bar{D}^T P_2 \bar{B}_2 (-\dot{D} - D) &\leq 2 \|\tilde{X}\| \|P_2 \bar{B}_2\| (\|\dot{D}\| + \|D\|) \\ &\leq 2 \|P_2 \bar{B}_2\|^2 \|\tilde{X}\|^2 + \|D\|^2 + \|\dot{D}\|^2 \\ &\leq L_0 \|\tilde{X}\|^2 + \|D\|^2 + \|\dot{D}\|^2 \leq L_0 \|\tilde{X}\|^2 + \theta_0^2 + \theta_1^2\end{aligned}\quad (26)$$

we have

$$\dot{V}_0 \leq -\tilde{X}^T (Q - \delta_0 I_{3n \times 3n}) \tilde{X} + \frac{1}{\lambda_1} \|\bar{\Theta}\|^2 + \theta_0^2 + \theta_1^2, \quad (27)$$

where $L_0 = 2\|P\bar{B}\|^2$, $L_4 = \lambda_1\|P\bar{B}\|^2\mu^2$, $\delta_0 = L_0 + L_4$, $\|\phi(\bar{Z})\| \leq \mu$, and $\lambda_1 > 0$ is a design constant.

3.2. Design of Output Feedback Using Backstepping Method.

Up to now, the state observer and the disturbance observer have been proposed for the uncertain robot manipulator using the RBFNN. Next, the adaptive neural output feedback control based on the developed disturbance observer will be proposed for uncertain robot manipulators with input saturation using the backstepping method and adaptation technique. The detailed design procedure is described as follows.

Step 1. To develop the adaptive neural output feedback control scheme, we define

$$e_1 = \hat{x}_1 - y_d \quad (28)$$

$$e_2 = \hat{x}_2 - \alpha_1 - \dot{y}_d, \quad (29)$$

where α_1 is a virtual control law which will be designed.

Considering (16) and differentiating e_1 with respect to time yield

$$\dot{e}_1 = \dot{\hat{x}}_1 - \dot{y}_d = \hat{x}_2 + (L_1 - L_2)(x_1 - \hat{x}_1) - \dot{y}_d. \quad (30)$$

Considering (29), we obtain

$$\dot{e}_1 = e_2 + \alpha_1 + (L_1 - L_2)(x_1 - \hat{x}_1). \quad (31)$$

The virtual control law α_1 is designed as

$$\alpha_1 = -K_1 e_1 - (L_1 - L_2)(x_1 - \hat{x}_1), \quad (32)$$

where $K_1 = K_1^T > 0$ is a design parameter.

Substituting (32) into (31), we have

$$\dot{e}_1 = -K_1 e_1 + e_2. \quad (33)$$

Consider the Lyapunov function candidate as

$$V_1 = \frac{1}{2} e_1^T e_1. \quad (34)$$

Invoking (33), the time derivative of V_1 is given by

$$\dot{V}_1 = -e_1^T K_1 e_1 + e_1^T e_2. \quad (35)$$

Step 2. Considering (16) and (29) and differentiating e_2 with respect to time yield

$$\begin{aligned} \dot{e}_2 &= \dot{\hat{x}}_2 - \dot{\alpha}_1 - \ddot{y}_d \\ &= \widehat{\Theta}^T \phi(\widehat{Z}) + G(x_1) \text{sat}(v) + \widehat{D} + L_2(x_1 - \hat{x}_1) \\ &\quad + L_2(\dot{x}_1 - \dot{\hat{x}}_1) - \dot{\alpha}_1 - \ddot{y}_d, \end{aligned} \quad (36)$$

where $\dot{\alpha}_1 = -K_1 \dot{e}_1 - (L_1 - L_2)(x_2 - \hat{x}_2 - (L_1 - L_2)(x_1 - \hat{x}_1))$.

Defining $\varsigma = -K_1 \dot{e}_1 + (L_1 - L_2)(L_1 - L_2)(x_1 - \hat{x}_1)$, we have

$$\dot{\alpha}_1 = \varsigma + (L_1 - L_2) \tilde{x}_2. \quad (37)$$

Invoking (17) and (37), (36) can be written as

$$\begin{aligned} \dot{e}_2 &= \widehat{\Theta}^T \phi(\widehat{Z}) + G(x_1) \text{sat}(v) + \widehat{D} + L_2(x_1 - \hat{x}_1) \\ &\quad + L_2(x_2 - \hat{x}_2) - L_2(L_1 - L_2)(x_1 - \hat{x}_1) - \varsigma \\ &\quad - (L_1 - L_2) \tilde{x}_2 - \ddot{y}_d. \end{aligned} \quad (38)$$

Considering $\Delta u = \text{sat}(v) - v$ and $\tilde{x}_2 = \hat{x}_2 - x_2$, (38) can be expressed as

$$\begin{aligned} \dot{e}_2 &= \widehat{\Theta}^T \phi(\widehat{Z}) + G(x_1)v + G(x_1)\Delta u + \widehat{D} \\ &\quad + L_2(x_1 - \hat{x}_1) - L_2\tilde{x}_2 - L_2(L_1 - L_2)(x_1 - \hat{x}_1) \\ &\quad - \varsigma - (L_1 - L_2) \tilde{x}_2 - \ddot{y}_d. \end{aligned} \quad (39)$$

Utilizing the outputs of the disturbance observer and the RBFNN, the adaptive neural output feedback control law v is designed as

$$\begin{aligned} v &= -G(x)^{-1} \\ &\quad \cdot (K_2 e_2 + \widehat{\Theta}^T \phi(\widehat{Z}) + \widehat{D} - \varsigma - \ddot{y}_d + e_1 + v_0) \\ v_0 &= L_2(x_1 - \hat{x}_1) - L_2(L_1 - L_2)(x_1 - \hat{x}_1) \\ &\quad + \text{sign}(e_2) g(x_1) \rho(\hat{x}) \widehat{\xi}, \end{aligned} \quad (40)$$

where $K_2 = K_2^T > 0$, $\text{sign}(e_2) = \text{diag}\{\text{sign}(e_{2i})\}$, with $\text{sign}(\cdot)$ denoting the sign function, $g(x_1) = [\|g_1(x_1)\|, \|g_2(x_1)\|, \dots, \|g_n(x_1)\|]^T$, with $g_i(x_1)$ denoting the i th row of input matrix $G(x_1)$, and $\widehat{\xi}$ is an estimate of unknown constant ξ .

Substituting (40) into (39), we obtain

$$\begin{aligned} \dot{e}_2 &= -K_2 e_2 - e_1 - \text{sign}(e_2) g(x_1) \rho(\hat{x}) \widehat{\xi} + G(x_1) \Delta u \\ &\quad - L_2 \tilde{x}_2 - (L_1 - L_2) \tilde{x}_2 \\ &= -K_2 e_2 - e_1 - \text{sign}(e_2) g(x_1) \rho(\hat{x}) \widehat{\xi} + G(x_1) \Delta u \\ &\quad - L_1 \tilde{x}_2. \end{aligned} \quad (41)$$

In accordance with (41), we have

$$\begin{aligned} e_2^T \dot{e}_2 &= -e_2^T K_2 e_2 - e_2^T e_1 - e_2^T \text{sign}(e_2) g(x_1) \rho(\hat{x}) \widehat{\xi} \\ &\quad + e_2^T G(x_1) \Delta u - e_2^T L_1 \tilde{x}_2. \end{aligned} \quad (42)$$

Considering $\|\Delta u\| \leq \rho(x)\xi$, we obtain

$$\begin{aligned} e_2^T \text{sign}(e_2) g(x_1) \rho(\hat{x}) \widehat{\xi} &= \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(\hat{x}) \widehat{\xi} \\ e_2^T G(x_1) \Delta u &= e_2^T [g_1(x_1) \Delta u, g_2(x_1) \Delta u, \dots, g_n(x_1) \Delta u]^T \\ &\leq \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(x) \xi. \end{aligned} \quad (43)$$

Substituting (43) into (42) yields

$$\begin{aligned} e_2^T \dot{e}_2 &\leq -e_2^T K_2 e_2 - e_2^T e_1 - \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(\hat{x}) \widehat{\xi} \\ &\quad + \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(x) \xi - e_2^T L_1 \tilde{x}_2 \\ &= -e_2^T K_2 e_2 - e_2^T e_1 - \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(\hat{x}) \widehat{\xi} \\ &\quad - e_2^T L_1 \tilde{x}_2 + \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| (\rho(x) - \rho(\hat{x})) \xi, \end{aligned} \quad (44)$$

where $\widetilde{\xi} = \widehat{\xi} - \xi$.

Considering Assumption 2, we can obtain $\|g_i(x_1)\| \leq \beta_0$ with $\beta_0 > 0$. Invoking Assumption 5, we have

$$\begin{aligned} &\sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| (\rho(x) - \rho(\hat{x})) \xi \\ &\leq l\beta_0 \|\xi\| \|\bar{x}\| \sum_{i=1}^n |e_{2i}| \\ &\leq \frac{1}{2} (l\beta_0 \|\xi\|)^2 \|e_2\|^2 + \frac{1}{2} \|\bar{x}\|^2. \end{aligned} \quad (45)$$

Consider the Lyapunov function candidate

$$V_2 = \frac{1}{2} e_2^T e_2 + \frac{1}{2\gamma} \widetilde{\xi}^T \widetilde{\xi} + \frac{1}{2} \text{tr}(\widetilde{\Theta}^T \Lambda^{-1} \widetilde{\Theta}), \quad (46)$$

where $\gamma > 0$ and $\Lambda = \Lambda^T > 0$ are design parameters.

Invoking (44), the time derivative of V_2 is

$$\begin{aligned} \dot{V}_2 &= e_2^T \dot{e}_2 + \frac{1}{\gamma} \widetilde{\xi}^T \dot{\widetilde{\xi}} + \text{tr}(\widetilde{\Theta}^T \Lambda^{-1} \dot{\widetilde{\Theta}}) \\ &\leq -e_2^T K_2 e_2 - e_2^T e_1 - \sum_{i=1}^n |e_{2i}| \|g_i(x_1)\| \rho(\hat{x}) \widehat{\xi} \\ &\quad + \frac{1}{2} (l\beta_0 \|\xi\|)^2 \|e_2\|^2 + \frac{1}{2} \|\bar{x}\|^2 - e_2^T L_1 \tilde{x}_2 \\ &\quad + \frac{1}{\gamma} \widetilde{\xi}^T \dot{\widetilde{\xi}} - \text{tr}(\widetilde{\Theta}^T \Lambda^{-1} \dot{\widetilde{\Theta}}). \end{aligned} \quad (47)$$

The parameter adaptation laws of $\hat{\xi}$ and $\hat{\Theta}$ are designed as

$$\begin{aligned}\dot{\hat{\xi}} &= \gamma (|e_{2i}| \|g_i(x_1)\| \rho^T(\hat{x}) - \sigma_1 \hat{\xi}) \\ \dot{\hat{\Theta}} &= \Lambda (\phi(\hat{Z}) e_2 + \sigma_2 \hat{\Theta}),\end{aligned}\quad (48)$$

where $\sigma_1 > 0$ and $\sigma_2 > 0$ are the design parameters.

Substituting (48) into (47) and considering $\tilde{\xi} = \hat{\xi} - \xi$ and $\tilde{\Theta} = \hat{\Theta} - \Theta^*$, we obtain

$$\begin{aligned}\dot{V}_2 &\leq -e_2^T K_2 e_2 - e_2^T e_1 - e_2^T L_1 \tilde{x}_2 - \sigma_1 \tilde{\xi} \tilde{\xi}^T - e_2^T \tilde{\Theta}^T \phi(\hat{Z}) \\ &\quad - \sigma_2 \text{tr}(\tilde{\Theta}^T \tilde{\Theta}) + \frac{1}{2} (\beta_0 \|\tilde{\xi}\|)^2 \|e_2\|^2 + \frac{1}{2} \|\tilde{X}\|^2.\end{aligned}\quad (49)$$

Considering the following facts

$$\begin{aligned}2\tilde{\xi} \tilde{\xi}^T &\geq \tilde{\xi}^2 - \xi^2 \\ -2e_2^T L_1 \tilde{x}_2 &\leq \|L_1\|^2 \|e_2\|^2 + \|\tilde{X}\|^2 \\ -2e_2^T \tilde{\Theta}^T \phi(\hat{Z}) &\leq \lambda_2 \mu^2 \|e_2\|^2 + \frac{1}{\lambda_2} \|\tilde{\Theta}\|^2 \\ 2\text{tr}(\tilde{\Theta}^T \tilde{\Theta}) &= \|\tilde{\Theta}\|^2 + \|\tilde{\Theta}\| - \|\Theta^*\| \geq \|\tilde{\Theta}\|^2 - \|\Theta^*\|\end{aligned}\quad (50)$$

we have

$$\begin{aligned}\dot{V}_2 &\leq -e_2^T (K_2 - \delta_1 I_{3n \times 3n}) e_2 - \frac{\sigma_1}{2} \tilde{\xi}^2 \\ &\quad - \left(\frac{\sigma_2}{2} - \frac{1}{2\lambda_2} \right) \|\tilde{\Theta}\|^2 - e_2^T e_1 + \|\tilde{X}\|^2 + \frac{\sigma_1}{2} \xi^2 \\ &\quad + \frac{1}{2} \|\Theta^*\|^2,\end{aligned}\quad (51)$$

where $\lambda_2 > 0$ is a design parameter and $\delta_1 = (1/2)\|L_1\|^2 + (1/2)\lambda_2 \mu^2 + (1/2)(\beta_0 \|\xi\|)^2$.

To analyze the convergence of the whole closed-loop system, the Lyapunov function candidate is considered as

$$V = \sum_{i=0}^2 V_i.\quad (52)$$

Considering (27), (35), and (51) yields

$$\begin{aligned}\dot{V} &\leq -\tilde{X}^T (Q - \delta_0 I_{3n \times 3n}) \tilde{X} + \frac{1}{\lambda_1} \|\tilde{\Theta}\|^2 + \theta_0^2 + \theta_1^2 \\ &\quad - e_1^T K_1 e_1 - e_2^T (K_2 - \delta_1 I_{3n \times 3n}) e_2 - \frac{\sigma_1}{2} \tilde{\xi}^2 \\ &\quad - \left(\frac{\sigma_2}{2} - \frac{1}{2\lambda_2} \right) \|\tilde{\Theta}\|^2 + \|\tilde{X}\|^2 + \frac{\sigma_1}{2} \xi^2 + \frac{1}{2} \|\Theta^*\|^2 \\ &= -\tilde{X}^T (Q - (\delta_0 + 1) I_{3n \times 3n}) \tilde{X} - e_1^T K_1 e_1 \\ &\quad - e_2^T (K_2 - \delta_1 I_{3n \times 3n}) e_2 - \frac{\sigma_1}{2} \tilde{\xi}^2 \\ &\quad - \left(\frac{\sigma_2}{2} - \frac{1}{\lambda_1} - \frac{1}{2\lambda_2} \right) \|\tilde{\Theta}\|^2 + \frac{\sigma_1}{2} \xi^2 + \frac{1}{2} \|\Theta^*\|^2 \\ &\quad + \theta_0^2 + \theta_1^2 \leq -\kappa V + C,\end{aligned}\quad (53)$$

where

$$\begin{aligned}\kappa &= \min \left(\begin{array}{l} \lambda_{\min}(Q - (\delta_0 + 1) I_{3n \times 3n}), 2\lambda_{\min}(K_1), \\ 2\lambda_{\min}(K_2 - \delta_1 I_{3n \times 3n}), \sigma_1, \frac{2(\sigma_2/2 - 1/\lambda_1 - 1/2\lambda_2)}{\lambda_{\max}(\Lambda)} \end{array} \right) \\ &> 0\end{aligned}\quad (54)$$

> 0

$$C = \frac{\sigma_1}{2} \xi^2 + \frac{1}{2} \|\Theta^*\|^2 + \theta_0^2 + \theta_1^2 > 0.$$

Integration of (53) yields

$$0 \leq V \leq \frac{C}{\kappa} + \left(V(0) - \frac{C}{\kappa} \right) e^{-\kappa t}.\quad (55)$$

The disturbance observer based adaptive neural output feedback control design procedure of the uncertain robot manipulator can be summarized in the following theorem, which includes the result of disturbance observer based adaptive neural output feedback control with unknown system uncertainty, time-varying external disturbance, and input saturation.

Theorem 9. Consider the robot manipulator system (1) with unknown time-varying external disturbance. The state observer (14) with (15) is designed for the robot manipulator. The unknown compounded disturbance is estimated using the disturbance observer (20). Using the outputs of the state observer, the disturbance observer, and the RBFNN, the adaptive neural output feedback control for the robot manipulator is designed as (40), with the parameter updated laws (48). Then, all closed-loop system signals are semiglobally uniformly bounded under the proposed disturbance observer based adaptive neural output feedback control scheme.

Proof. According to (55), we can obtain that V is exponentially convergent, and $\lim_{t \rightarrow \infty} V = C/\kappa$. Thus, all signals of the closed-loop system are uniformly ultimately bounded. Due to bounded e_1 and \tilde{x}_1 , we can know that the tracking error $y - y_d$ is bounded under the proposed adaptive output feedback control scheme from (28) and (55). This concludes the proof. \square

4. Simulation Results

To illustrate the effectiveness of the developed adaptive neural output feedback control scheme based on the disturbance observer, numerical simulation results are presented for the robot manipulator. The dynamic equations of a two-link robotic manipulator can be written as [63]

$$\begin{aligned}&\begin{bmatrix} a_{11}(q_2) & a_{12}(q_2) \\ a_{12}(q_2) & a_{22}(q_2) \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} \\ &= \begin{bmatrix} \beta_{12} \dot{q}_1^2 + 2\beta_{12}(q_2) \dot{q}_1 \dot{q}_2 \\ -\beta_{12}(q_2) \dot{q}_2^2 \end{bmatrix} + \begin{bmatrix} \gamma_{01}(q_1, q_2) g \\ \gamma_{02}(q_1, q_2) g \end{bmatrix} \\ &\quad + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix},\end{aligned}\quad (56)$$

TABLE I: Parameters of the two-link robotic manipulator.

Parameter	Value
r_1	1 m
r_2	0.8 m
J_1	5 kg·m
J_2	5 kg·m
m_1	0.5 kg
m_2	1.5 kg

where $a_{11}(q_2) = (m_1 + m_2)r_1^2 + m_2r_2^2 + 2m_2r_1r_2 \cos(q_2) + J_1$, $a_{12}(q_2) = m_2r_2^2 + m_2r_1r_2 \cos(q_2)$, $a_{22} = m_2r_2^2 + J_2$, $\beta_{12} = m_2r_1r_2 \sin(q_2)$, $\gamma_{01}(q_1, q_2) = -((m_1 + m_2)r_1 \cos(q_2) + m_2r_2 \cos(q_1 + q_2))$, $\gamma_{02}(q_1, q_2) = -m_2r_2 \cos(q_1 + q_2)$, and d_i ($i = 1, 2$) represent the unknown time-varying disturbances of the robot manipulator system. We assume that $M_0(q) = \begin{bmatrix} a_{11}(q_2) & a_{12}(q_2) \\ a_{12}(q_2) & a_{22}(q_2) \end{bmatrix}$ is the known nominal symmetric positive-definite inertia matrix and $\Delta M(q) = 0.15M_0(q)$ is the uncertainty of the symmetric positive-definite inertia matrix. All parameters of the two-link robotic manipulator are given in Table I [63].

In the simulation study, the state observer is designed as (14) and (15). The compounded disturbance is estimated using the disturbance observer which is proposed as (20). Using the output of the state observer, the disturbance observer, and the RBFNN, the adaptive neural output feedback control of the robot manipulator is designed as (40) and the parameter updated laws are chosen as (48).

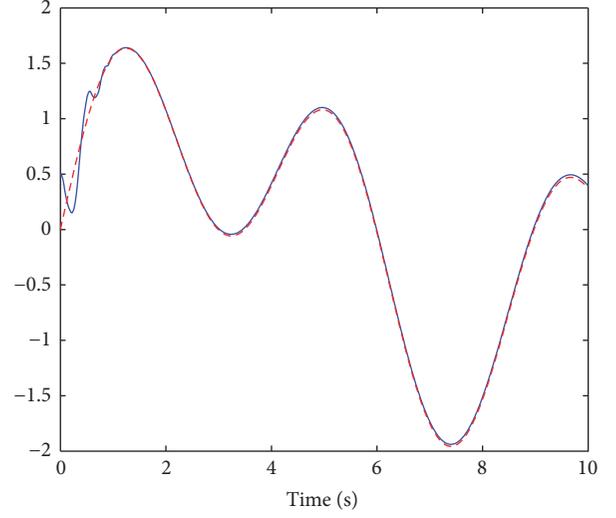
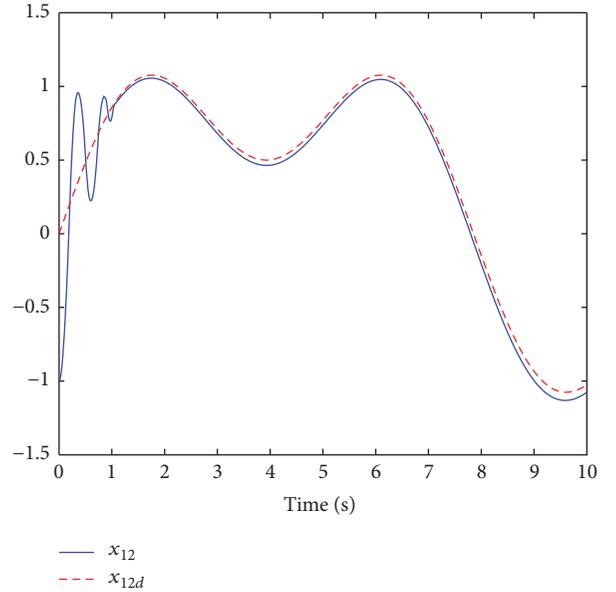
Since the sign function term $\text{sign}(e_{2i})$ in the adaptive neural output feedback control design will introduce a high-frequency oscillation into the system that can excite the unmodeled dynamics, to avoid chattering, function $\text{sign}(e_{2i})$ is replaced with $\text{sat}(e_{2i}/\tau_i)$ in the simulation, where τ_i is the boundary layer width and $\text{sat}(e_{2i}/\tau_i)$ is given by [68]

$$\text{sat}\left(\frac{e_{2i}}{\tau}\right) = \begin{cases} \frac{e_{2i}}{\tau}, & \left|\frac{e_{2i}}{\tau}\right| \leq 1 \\ \text{sign}\left(\frac{e_{2i}}{\tau_i}\right), & \left|\frac{e_{2i}}{\tau_i}\right| > 1. \end{cases} \quad (57)$$

The initial condition of the updated parameters is chosen as $\widehat{W} = \text{diag}\{0.1\}$ and $\widehat{\theta} = [0, 0]^T$. All design parameters of the disturbance observer based adaptive neural output feedback control are chosen as $L_1 = \text{diag}\{200\}$, $L_2 = \text{diag}\{300\}$, $L_3 = \text{diag}\{100\}$, $\gamma = 0.2$, $\sigma_1 = 0.2$, $\Lambda = \text{diag}\{2\}$, $\sigma_2 = 0.5$, $\tau_i = 0.2$, $K_1 = \text{diag}\{120\}$, and $L_2 = \text{diag}\{135\}$.

To illustrate the effectiveness of the proposed disturbance observer based adaptive neural output feedback control scheme, the tracking control simulation results are presented for the time-varying desired tracking signals.

The desired trajectories are taken as $x_{11d} = 0.6 \cos(0.6t) + 1.5 \cos(1.5t)$ and $x_{12d} = 0.4 \cos(0.4t) + 0.6 \cos(1.2t)$ and the unknown disturbances are assumed as $D_1 = 2 \sin(t)$ and $D_2 = 1.2 \cos(1.5t)$. The initial states are chosen as $x_1 = [0.5, -1]^T$ and $x_2 = [0.1, 0.1]^T$.

FIGURE 1: Tracking result of x_{11} in response to x_{11d} .FIGURE 2: Tracking result of x_{12} in response to x_{12d} .

The tracking control results of the two-link robotic manipulator are shown in Figures 1, 2, and 3 under the proposed adaptive neural output feedback control scheme. From Figures 1–3, for the time-varying desired trajectories, we note that x_{11} and x_{12} can track the corresponding desired signals x_{11d} and x_{12d} with a small tracking error under the integrated system uncertainty, consisting of the unmeasured system state, the unknown external disturbance, and the input saturation. At the same time, the state estimate results of two joints are given in Figures 4–7. In accordance with the simulation results of state observer, we can see that the state

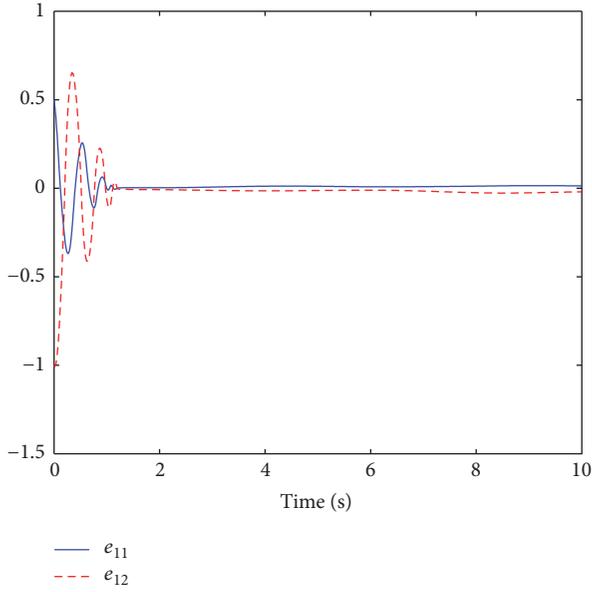


FIGURE 3: Tracking errors to the two desired trajectories x_{11d} and x_{12d} .

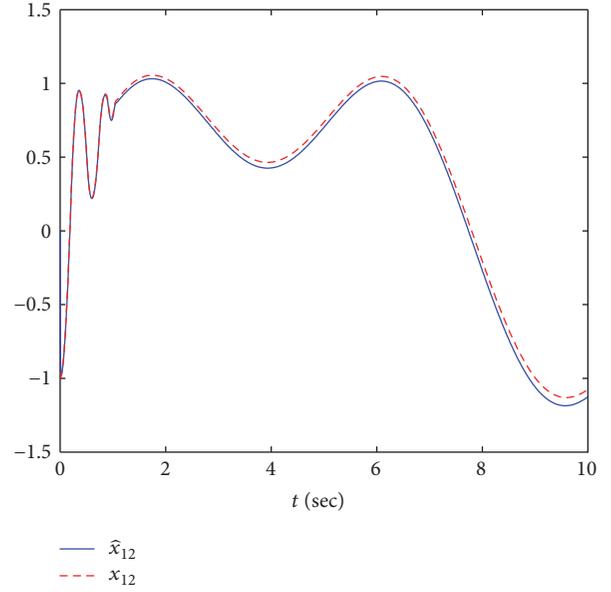


FIGURE 5: Responses of x_{12} and its estimate.

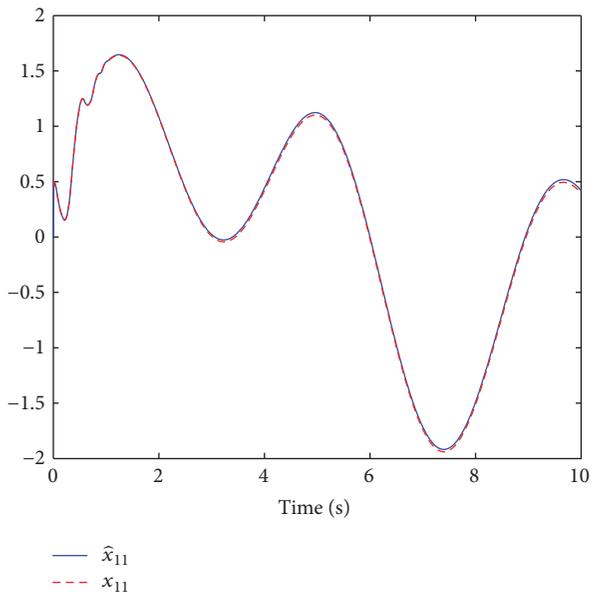


FIGURE 4: Responses of x_{11} and its estimate.

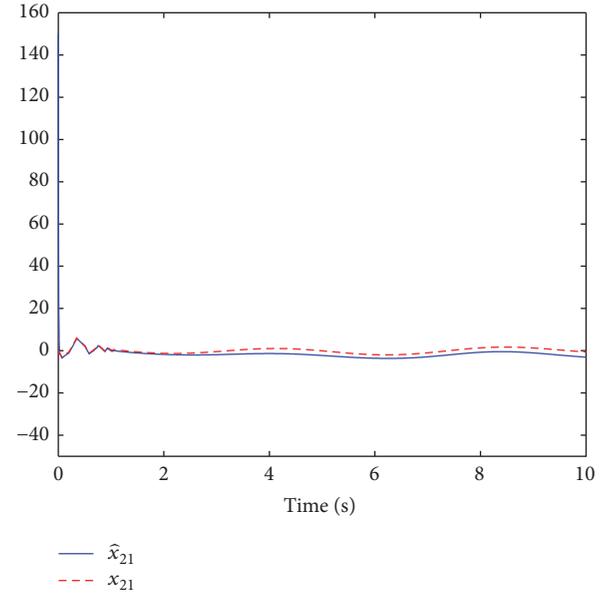


FIGURE 6: Responses of x_{21} and its estimate.

estimate performance is satisfactory and the estimate errors are asymptotically convergent. According to the simulation results presented in Figures 8 and 9, we can also observe that the control input of the developed disturbance observer based adaptive neural output feedback control is bounded and the saturation appears in the initial control stage. However, the saturation phenomenon is ultimately eliminated via the adaptation method under our proposed adaptive output feedback control scheme.

Based on the simulation results of above simulation case, we can conclude that the developed disturbance observer based adaptive neural output feedback control is effective

for the two-link robotic manipulator with the time-varying external disturbance, the system uncertainty, and input saturation for the desired bounded time-varying tracking trajectories.

5. Conclusion

Disturbance observer based adaptive neural output feedback tracking control has been proposed for the robot manipulator, which is subject to system uncertainty, unknown time-varying external disturbance, and input saturation. The state observer is developed to estimate unmeasured system

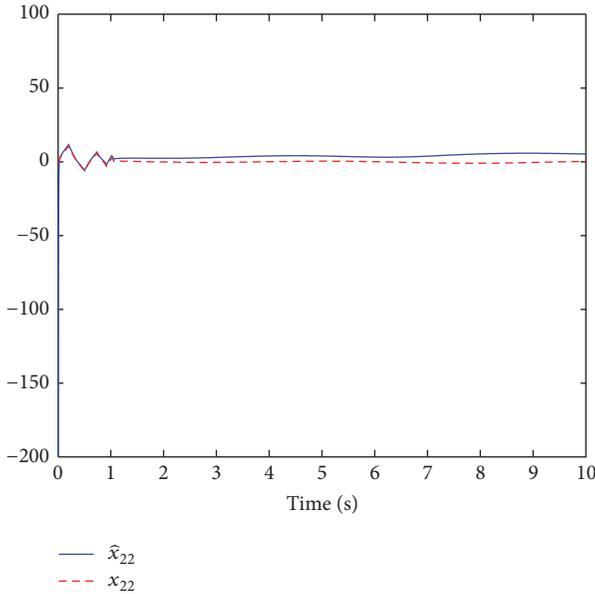


FIGURE 7: Responses of x_{22} and its estimate.

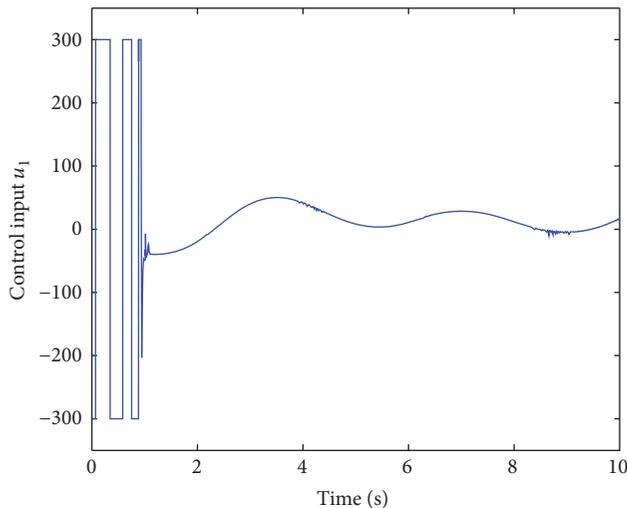


FIGURE 8: Response of the control input u_1 .

states. To improve the ability of disturbance attenuation and the tracking robustness, the disturbance observer has been developed to estimate the combination of the neural network approximation error and the unknown time-varying external disturbance. Based on the outputs of the developed state observer, disturbance observer, and the radial basis function neural network, the adaptive neural output feedback control scheme has been proposed for the robot manipulator. The asymptotical convergence of all closed-loop signals has been proved using rigorous Lyapunov analysis. Simulation results confirm the effectiveness of the proposed disturbance observer based adaptive neural output feedback control scheme.

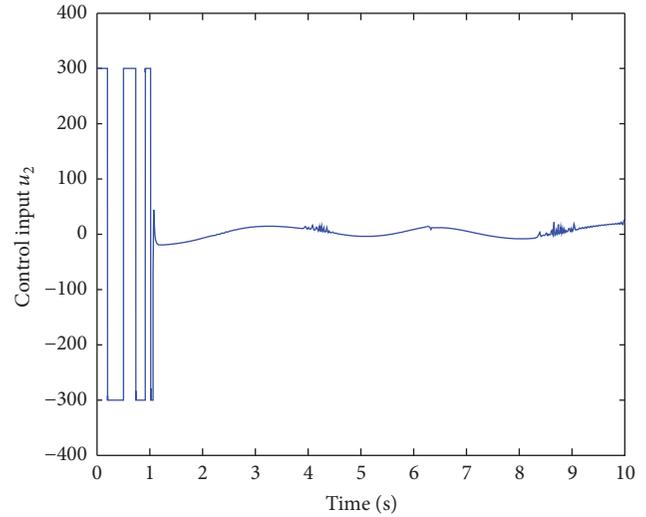


FIGURE 9: Response of the control input u_2 .

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] M. Chen, B. Jiang, and R. X. Cui, "Robust control for rigid robotic manipulators using nonlinear disturbance observer," *International Journal of Robotics and Automation*, vol. 29, no. 3, pp. 305–311, 2014.
- [2] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by elm-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 8, pp. 2398–2409, 2017.
- [3] Y. Kang, Z. Li, X. Cao, and D. Zhai, "Robust control of motion/force for robotic manipulators with random time delays," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 5, pp. 1708–1718, 2013.
- [4] J. P. Kolhe, M. Shaheed, T. S. Chandar, and S. E. Talole, "Robust control of robot manipulators based on uncertainty and disturbance estimation," *International Journal of Robust and Nonlinear Control*, vol. 23, no. 1, pp. 104–122, 2013.
- [5] C. Yang, J. Luo, Y. Pan, Z. Liu, and C. Su, "Personalized variable gain control with tremor attenuation for robot teleoperation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2017.
- [6] R.-J. Wai and R. Muthusamy, "Fuzzy-neural-network inherited sliding-mode control for robot manipulator including actuator dynamics," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 2, pp. 274–287, 2013.
- [7] S. P. Chan, "A disturbance observer for robot manipulators with application to electronic components assembly," *IEEE Transactions on Industrial Electronics*, vol. 42, no. 5, pp. 487–493, 1995.
- [8] J.-J. Slotine and S. S. Sastry, "Tracking control of nonlinear systems using sliding surfaces, with application to robot manipulators," *International Journal of Control*, vol. 38, no. 2, pp. 465–492, 1983.
- [9] R.-J. Wai and P.-C. Chen, "Intelligent tracking control for robot manipulator including actuator dynamics via TSK-type fuzzy

- neural network," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 4, pp. 552–559, 2004.
- [10] W. Dong, "On trajectory and force tracking control of constrained mobile manipulators with parameter uncertainty," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 38, no. 9, pp. 1475–1484, 2002.
- [11] Z. Man and M. Palaniswami, "A robust tracking control scheme for rigid robotic manipulators with uncertain dynamics," *Computers and Electrical Engineering*, vol. 21, no. 3, pp. 211–220, 1995.
- [12] L. Cheng, Z.-G. Hou, and M. Tan, "Adaptive neural network tracking control for manipulators with uncertain kinematics, dynamics and actuator model," *Automatica*, vol. 45, no. 10, pp. 2312–2318, 2009.
- [13] M. Chen and G. Tao, "Adaptive fault-tolerant control of uncertain nonlinear large-scale systems with unknown dead zone," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1851–1862, 2016.
- [14] J. I. Mulero-Martinez, "Robust GRBF static neurocontroller with switch logic for control of robot manipulators," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 7, pp. 1053–1064, 2012.
- [15] M. Chen, G. Tao, and B. Jiang, "Dynamic surface control using neural networks for a class of uncertain nonlinear systems with input saturation," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 9, pp. 2086–2097, 2015.
- [16] M. K. Ciliz, "Adaptive control of robot manipulators with neural network based compensation of frictional uncertainties," *Robotica*, vol. 23, no. 2, pp. 159–167, 2005.
- [17] R.-J. Wai, "Tracking control based on neural network strategy for robot manipulator," *Neurocomputing*, vol. 51, pp. 425–445, 2003.
- [18] M. J. Er and Y. Gao, "Robust adaptive control of robot manipulators using generalized fuzzy neural networks," *IEEE Transactions on Industrial Electronics*, vol. 50, no. 3, pp. 620–628, 2003.
- [19] R.-J. Wai and P.-C. Chen, "Robust neural-fuzzy-network control for robot manipulator including actuator dynamics," *IEEE Transactions on Industrial Electronics*, vol. 53, no. 4, pp. 1328–1349, 2006.
- [20] K. W. Lee and H. K. Khalil, "Adaptive output feedback control of robot manipulators using high-gain observer," *International Journal of Control*, vol. 67, no. 6, pp. 869–886, 1997.
- [21] S. C. Tong, T. Wang, Y. M. Li, and B. Chen, "A combined backstepping and stochastic small-gain approach to robust adaptive fuzzy output feedback control," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 2, pp. 314–327, 2013.
- [22] M. Chen, S. S. Ge, B. V. E. How, and Y. S. Choo, "Robust adaptive position mooring control for marine vessels," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 2, pp. 395–409, 2013.
- [23] Y. Liu, S. Tong, and C. L. P. Chen, "Adaptive fuzzy control via observer design for uncertain nonlinear systems with unmodeled dynamics," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 2, pp. 275–288, 2013.
- [24] S. Tong, Y. Li, and P. Shi, "Observer-based adaptive fuzzy backstepping output feedback control of uncertain MIMO pure-feedback nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 4, pp. 771–785, 2012.
- [25] M. Chen and S. Ge, "Adaptive neural output feedback control of uncertain nonlinear systems with unknown hysteresis using disturbance observer," *IEEE Transactions on Industrial Electronics*, vol. 26, no. 12, pp. 7706–7716, 2015.
- [26] Q. Zhou, P. Shi, S. Xu, and H. Li, "Adaptive output feedback control for nonlinear time-delay systems by fuzzy approximation approach," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 2, pp. 301–313, 2013.
- [27] F. Alonge, F. D'Ippolito, and F. M. Raimondi, "An adaptive control law for robotic manipulator without velocity feedback," *Control Engineering Practice*, vol. 11, no. 9, pp. 999–1005, 2003.
- [28] Y. H. Kim and F. L. Lewis, "Neural network output feedback control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 301–309, 1999.
- [29] W. Zeng and C. Wang, "Learning from NN output feedback control of robot manipulators," *Neurocomputing*, vol. 125, pp. 172–182, 2014.
- [30] P. R. Pagilla and M. Tomizuka, "An adaptive output feedback controller for robot arms: stability and experiments," *Automatica. A Journal of IFAC, the International Federation of Automatic Control*, vol. 37, no. 7, pp. 983–995, 2001.
- [31] M. Chen, "Disturbance attenuation tracking control for wheeled mobile robots with skidding and slipping," *IEEE Transactions on Industrial Electronics*, vol. 64, no. 4, pp. 3359–3368, 2017.
- [32] M. Chen and B. Jiang, "Robust attitude control of near space vehicles with time-varying disturbances," *International Journal of Control, Automation and Systems*, vol. 11, no. 1, pp. 182–187, 2013.
- [33] M. Chen, B. Ren, Q. Wu, and C. Jiang, "Anti-disturbance control of hypersonic flight vehicles with input saturation using disturbance observer," *Science China. Information Sciences*, vol. 58, no. 7, 070202, 12 pages, 2015.
- [34] M. Chen, P. Shi, and C. C. Lim, "Adaptive neural fault-tolerant control of a 3-dof model helicopter system," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 2, pp. 260–270, 2015.
- [35] X. Wei and L. Guo, "Composite disturbance-observer-based control and H_1 control for complex continuous models," *International Journal of Robust and Nonlinear Control*, vol. 20, no. 1, pp. 106–118, 2010.
- [36] L. Guo and W.-H. Chen, "Disturbance attenuation and rejection for systems with nonlinearity via DOBC approach," *International Journal of Robust and Nonlinear Control*, vol. 15, no. 3, pp. 109–125, 2005.
- [37] J. Yang, S. Li, and X. Yu, "Sliding-mode control for systems with mismatched uncertainties via a disturbance observer," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 1, pp. 160–169, 2013.
- [38] W.-H. Chen, "Nonlinear disturbance observer-enhanced dynamic inversion control of missiles," *Journal of Guidance, Control, and Dynamics*, vol. 26, no. 1, pp. 161–166, 2003.
- [39] W.-H. Chen, D. J. Ballance, P. J. Gawthrop, and J. O'Reilly, "A nonlinear disturbance observer for robotic manipulators," *IEEE Transactions on Industrial Electronics*, vol. 47, no. 4, pp. 932–938, 2000.
- [40] Z.-J. Yang, Y. Fukushima, and P. Qin, "Decentralized adaptive robust control of robot manipulators using disturbance observers," *IEEE Transactions on Control Systems Technology*, vol. 20, no. 5, pp. 1357–1365, 2012.
- [41] A. Mohammadi, M. Tavakoli, H. J. Marquez, and F. Hashemzadeh, "Nonlinear disturbance observer design for robotic manipulators," *Control Engineering Practice*, vol. 21, no. 3, pp. 253–267, 2013.

- [42] A. Nikoobin and R. Haghghi, "Lyapunov-based nonlinear disturbance observer for serial n-Link robot manipulators," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 55, no. 2-3, pp. 135–153, 2009.
- [43] M. Chen, P. Shi, and C.-C. Lim, "Robust constrained control for MIMO nonlinear systems based on disturbance observer," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 60, no. 12, pp. 3281–3286, 2015.
- [44] M. Chen, S. S. Ge, and B. V. E. How, "Robust adaptive neural network control for a class of uncertain MIMO nonlinear systems with input nonlinearities," *IEEE Transactions on Neural Networks*, vol. 21, no. 5, pp. 796–812, 2010.
- [45] M. Chen, S. S. Ge, and B. Ren, "Adaptive tracking control of uncertain MIMO nonlinear systems with input constraints," *Automatica*, vol. 47, no. 3, pp. 452–465, 2011.
- [46] B. Xiao, Q. Hu, and P. Shi, "Attitude stabilization of spacecrafts under actuator saturation and partial loss of control effectiveness," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 6, pp. 2251–2263, 2013.
- [47] M. Chen, "Constrained control allocation for overactuated aircraft using a neurodynamic model," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 12, pp. 1630–1641, 2016.
- [48] Y. Li, S. Tong, and T. Li, "Composite adaptive fuzzy output feedback control design for uncertain nonlinear strict-feedback systems with input saturation," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2299–2308, 2015.
- [49] M. Chen and S. S. Ge, "Direct adaptive neural control for a class of uncertain nonaffine nonlinear systems based on disturbance observer," *IEEE Transactions on Cybernetics*, vol. 43, no. 4, pp. 1213–1225, 2013.
- [50] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2017.
- [51] M. Chen, B. Jiang, and R. Cui, "Actuator fault-tolerant control of ocean surface vessels with input saturation," *International Journal of Robust and Nonlinear Control*, vol. 26, no. 3, pp. 542–564, 2016.
- [52] M. Chen, S. Shao, P. Shi, and Y. Shi, "Disturbance-observer-based robust synchronization control for a class of fractional-order chaotic systems," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 64, no. 4, pp. 417–421, 2017.
- [53] J. E. Slotine and M. W. Spong, "Robust robot control with bounded input torques," *Journal of Robotic Systems*, vol. 2, no. 4, pp. 329–352, 1985.
- [54] M. W. Spong, J. S. Thorp, and J. M. Kleinwaks, "The control of robot manipulators with bounded input," *IEEE Transactions on Automatic Control*, vol. 31, no. 6, pp. 483–490, 1986.
- [55] H.-S. Choi, "Robust control of robot manipulators with torque saturation using fuzzy logic," *Robotica*, vol. 19, no. 6, pp. 631–639, 2001.
- [56] H. Liu, K. Hao, and X. Lai, "Fuzzy saturated output feedback tracking control for robot manipulators: a singular perturbation theory based approach," *International Journal of Advanced Robotic Systems*, vol. 8, no. 4, pp. 43–53, 2011.
- [57] H. S. Liu, S. Q. Zhu, and Z. W. Chen, "Saturated output feedback tracking control for robot manipulators via fuzzy self-tuning," *Journal of Zhejiang University-SCIENCE C*, vol. 11, no. 12, pp. 956–966, 2010.
- [58] K. S. Eom, I. H. Suh, and W. K. Chung, "Disturbance observer based path tracking control of robot manipulator considering torque saturation," *Mechatronics*, vol. 11, no. 3, pp. 325–343, 2001.
- [59] C.-S. Chiu, K.-Y. Lian, and T.-C. Wu, "Robust adaptive motion/force tracking control design for uncertain constrained robot manipulators," *Automatica*, vol. 40, no. 12, pp. 2111–2119, 2004.
- [60] Y. M. Li, S. C. Tong, and T. S. Li, "Hybrid fuzzy adaptive output feedback control design for uncertain MIMO nonlinear systems with time-varying delays and input saturation," *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 4, pp. 841–853, 2016.
- [61] C.-Y. Su and Y. Stepanenko, "Backstepping-based hybrid adaptive control of robot manipulators incorporating actuator dynamics," *International Journal of Adaptive Control and Signal Processing*, vol. 11, no. 2, pp. 141–153, 1997.
- [62] Q. Hu, L. Xu, and A. Zhang, "Adaptive backstepping trajectory tracking control of robot manipulator," *Journal of the Franklin Institute. Engineering and Applied Mathematics*, vol. 349, no. 3, pp. 1087–1105, 2012.
- [63] M. Zhihong, A. P. Paplinski, and H. R. Wu, "A robust MIMO terminal sliding mode control scheme for rigid robotic manipulators," *IEEE Transactions on Automatic Control*, vol. 39, no. 12, pp. 2464–2469, 1994.
- [64] S. S. Ge and C. Wang, "Adaptive neural control of uncertain MIMO nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 15, no. 3, pp. 674–692, 2004.
- [65] Y. Feng, X. Yu, and Z. Man, "Non-singular terminal sliding mode control of rigid manipulators," *Automatica. A*, vol. 38, no. 12, pp. 2159–2167, 2002.
- [66] K. P. Tee and S. S. Ge, "Control of fully actuated ocean surface vessels using a class of feedforward approximators," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 4, pp. 750–756, 2006.
- [67] S. S. Ge and C. Wang, "Direct adaptive NN control of a class of nonlinear systems," *IEEE Transactions on Neural Networks*, vol. 13, no. 1, pp. 214–221, 2002.
- [68] M. Chen and W.-H. Chen, "Sliding mode control for a class of uncertain nonlinear system based on disturbance observer," *International Journal of Adaptive Control and Signal Processing*, vol. 24, no. 1, pp. 51–64, 2010.

Research Article

General Recurrent Neural Network for Solving Generalized Linear Matrix Equation

Zhan Li, Hong Cheng, and Hongliang Guo

School of Automation Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

Correspondence should be addressed to Hong Cheng; hcheng@uestc.edu.cn

Received 13 March 2017; Accepted 19 April 2017; Published 31 July 2017

Academic Editor: Yanan Li

Copyright © 2017 Zhan Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This brief proposes a general framework of the nonlinear recurrent neural network for solving online the generalized linear matrix equation (GLME) with global convergence property. If the linear activation function is utilized, the neural state matrix of the nonlinear recurrent neural network can globally and exponentially converge to the unique theoretical solution of GLME. Additionally, as compared with the case of using the linear activation function, two specific types of nonlinear activation functions are proposed for the general nonlinear recurrent neural network model to achieve superior convergence. Illustrative examples are shown to demonstrate the efficacy of the general nonlinear recurrent neural network model and its superior convergence when activated by the aforementioned nonlinear activation functions.

1. Introduction

Solving the generalized linear matrix equation (GLME) and its variants is an important issue which is widely encountered in scientific and engineering areas (e.g., feedback control system design [1], smart antenna array processing [2]). The well-known Lyapunov equation and Sylvester equation can be regarded as main special cases of GLME with reduced numbers of coefficients and variable matrices, which have drawn widespread interest of researchers and engineers in the past decades [3–7]. Without loss of generality, in this brief, the GLME problem is formulated as the following form:

$$\sum_{l=1}^p A_l X B_l = C, \quad (1)$$

where $A_l \in R^{m \times m}$, $B_l \in R^{n \times n}$, and $C \in R^{m \times n}$ denote coefficient matrices and $X \in R^{m \times n}$ denotes the unknown matrix to be obtained. Usually, it is complicated to analyze what circumstances the solution of (1) would be under in the traditional numerical way. To guarantee such GLME (1) solvable with the unique theoretical solution, coefficient matrices $A_l \in R^{m \times m}$ and $B_l \in R^{n \times n}$ can be practically configured with their eigenvalues all being positive or negative simultaneously. In many cases, the number of solutions of

(1) can be multiple or even none, depending on what kind of combinations matrices A_l and B_l would make to associate with the unknown matrix X . A lot of conventional serial approaches may be not efficient enough to solve online GLME due to their inherent drawbacks, and parallel computational approaches seem more preferable [8–13].

Regarded as another promising approach for parallel computation, dynamic neural networks based on analog solvers have been exploited comprehensively in computational intelligence fields [12, 14–16]. Different from a number of conventional numerical methods, approaches based on dynamic neural networks can be more realizable on specific parallel and distributed software or/and hardware architectures [17, 18]. This could highly enlarge utility of current neural networks towards various potential application domains towards high-performance computing. One basic type of dynamic neural networks, recurrent neural networks, which is analogous to the natural transient and steady process, has been applied in online parallel computing tasks with large-scale analog/digital circuit prototypes [19].

Our main contribution in this brief is to develop a general framework of recurrent neural network model to solve GLME (1). Since nonlinear phenomena occur frequently in neural network hardware implementation [19], the proposed general nonlinear framework may be more suitable for analog-based

computation. The neural state of the general recurrent neural network can globally converge to the theoretical solutions. If the general recurrent neural network is activated by the linear function, the exponential convergence can be achieved. On the other hand, certain nonlinear forms of such general neural network may be able to obtain more accurate solutions and faster convergence as compared with its linear model, so we propose two specific nonlinear activation functions for the general recurrent neural network model to achieve superior performance to solve GLME (1).

2. General Recurrent Neural Network Solver

In this section, we present and analyze the general model of recurrent neural network to solve GLME (1). If such model is activated by the linear function, state matrix $X(t) \in R^{m \times n}$ of the general recurrent neural network can globally and exponentially converge to the unique theoretical solution $X^* \in R^{m \times n}$. By exploiting specific nonlinear odd monotonically increasing activation functions, superior convergence is expected to be achieved. In the ensuing subsections, we will discuss their convergence properties of the general nonlinear recurrent neural network model together with its linear form.

2.1. General Nonlinear Neural Network Model. In this brief, the general nonlinear recurrent neural network model is proposed to solve GLME (1) as follows:

$$\dot{X}(t) = -\gamma \sum_{r=1}^p \left[A_r^T \mathcal{F} \left(\sum_{s=1}^p A_s X(t) B_s - C \right) B_r^T \right], \quad (2)$$

where operator $\mathcal{F}(\cdot) : R^{m \times n} \rightarrow R^{m \times n}$ denotes a nonlinear activation function array, with its each scalar-valued mapping unit $f(\cdot) : R \rightarrow R$ being a monotonically increasing odd activation function, and subscript $(\cdot)^T$ denotes transpose of matrix/vector. Such recurrent neural network model (2) can be generalized as the extended nonlinear version of recurrent neural network in [16] and the ensuing linear model. For the general nonlinear recurrent neural network model (2), we would have the following theorem.

Theorem 1. *The neural state matrix $X(t) \in R^{m \times n}$ of general nonlinear recurrent neural network model (2), starting from any initial value $X(0) \in R^{m \times n}$, can globally converge to the theoretical solution(s) $X^* \in R^{m \times n}$ of GLME (1).*

Proof. Firstly, we define the distance between the neural state and the theoretical solution as $\tilde{X}(t) = X(t) - X^* \in R^{m \times n}$. Accordingly, by substituting $X(t) = \tilde{X}(t) + X^* \in R^{m \times n}$ into neural network model (2), it can be further equivalently transformed as

$$\dot{\tilde{X}}(t) = -\gamma \sum_{r=1}^p A_r^T \mathcal{F} \left(\sum_{s=1}^p A_s \tilde{X}(t) B_s \right) B_r^T. \quad (3)$$

Next, the corresponding Lyapunov-function candidate is defined as follows:

$$V(\tilde{X}(t), t)_{(3)} = \|\tilde{X}(t)\|_F^2 = \|\text{vec}(\tilde{X}(t))\|_2^2, \quad (4)$$

where operators $\|\cdot\|_F$, $\|\cdot\|_2$, and \otimes , respectively, denote Frobenius norm of matrix, two norms of vector, and Kronecker product between matrices and $\text{vec}(\tilde{X}(t)) \in R^{mn}$ generates a new column vector obtained by stacking all column vectors of $\tilde{X}(t) \in R^{m \times n}$ together.

The time derivative of $V(\tilde{X}(t), t)_{(3)}$ is

$$\dot{V}(\tilde{X}(t), t)_{(3)} = 2 \text{vec}(\tilde{X}(t))^T \text{vec}(\dot{\tilde{X}}(t)). \quad (5)$$

Considering the following vectorization equality based on (2),

$$\begin{aligned} \text{vec}(\dot{\tilde{X}}(t)) &= -\gamma \left(\sum_{r=1}^p B_r \otimes A_r^T \right) \\ &\cdot \mathcal{F} \left(\left(\sum_{s=1}^p B_s^T \otimes A_s \right) \text{vec}(\tilde{X}(t)) \right), \end{aligned} \quad (6)$$

we can further derive (5) as

$$\begin{aligned} \dot{V}(\tilde{X}(t), t)_{(3)} &= -2\gamma \text{vec}(\tilde{X}(t))^T \left(\sum_{r=1}^p B_r \otimes A_r^T \right) \\ &\cdot \mathcal{F} \left(\left(\sum_{s=1}^p B_s^T \otimes A_s \right) \text{vec}(\tilde{X}(t)) \right) \\ &= -2\gamma \left[\left(\sum_{l=1}^p B_l^T \otimes A_l \right) \text{vec}(\tilde{X}(t)) \right]^T \\ &\cdot \mathcal{F} \left(\left(\sum_{l=1}^p B_l^T \otimes A_l \right) \text{vec}(\tilde{X}(t)) \right). \end{aligned} \quad (7)$$

For nonlinear activation function array $\mathcal{F}(\cdot)$, its individual scalar-valued entry $f(\cdot)$ is odd and monotonically increasing, which can guarantee

$$uf(u) = \begin{cases} > 0, & u \in R, u \neq 0; \\ = 0, & u \in R, u = 0. \end{cases} \quad (8)$$

Thus, $\dot{V}(\tilde{X}(t), t)_{(3)} \leq 0$, which implies that $\tilde{X}(t) \in R^{m \times n}$ could globally converge to zero matrix $\mathbf{0} \in R^{m \times n}$ according to Lyapunov theory [20]; that is, state matrix $X(t) \in R^{m \times n}$ of (2) globally converges to the theoretical solution(s) $X^* \in R^{m \times n}$ of GLME (1). All of these above complete the proof. \square

According to Theorem 1, the general nonlinear neural network model (2) can be activated by a number of odd monotonically increasing functions to solve GLME (1) which is with existent theoretical solutions (unique or multiple), which will broadly enlarge the utility domain of (2) towards manifold model generation. As we may know, nonlinear elements are frequently encountered in analog/digital circuit prototypes of neural networks [19, 21]; involving nonlinear activation functions can be beneficial to potential design and implication. On the other hand, faster convergence is indeed required for solving GLME (1) when the linear model

might not satisfy increasing computational requirements. With expectancy, the nonlinear neural network model (2) can attain superior convergence to that of (10) if proper activation functions are exploited. Before inducing the superior nonlinear-function activated models, we herein address the linear model of the general nonlinear recurrent neural network and discuss its convergence property.

2.2. Linear Neural Network Model. To solve GLME (1), we firstly define a scalar-valued error function $E(t) = \|\sum_{l=1}^p A_l X(t) B_l - C\|_F^2 / 2 \in [0, +\infty)$ associated with (1), where operator $\|\cdot\|_F$ denotes the Frobenius norm. In order to eliminate error function $E(t)$ to zero as t increases, gradient-descent manner is adopted:

$$\dot{X}(t) = -\gamma \frac{\partial E(t)}{\partial X(t)}, \quad (9)$$

where design parameter $\gamma > 0$ scales the convergence rate.

According to preliminaries on matrix-differential theory [22], (9) is further expanded to the following dynamic form:

$$\dot{X}(t) = -\gamma \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s X(t) B_s B_r^T + \gamma \sum_{l=1}^p A_l^T C B_l^T. \quad (10)$$

For linear model (10), we would have the following theorem.

Theorem 2. *If the linear neural network model (10) is employed to solve GLME (1), starting from initial condition $X(0) \in R^{m \times n}$, the state matrix $X(t) \in R^{m \times n}$ of (10) can globally exponentially converge to unique theoretical solution $X^* \in R^{m \times n}$.*

Proof. Using the transformation $\tilde{X}(t) = X(t) - X^* \in R^{m \times n}$ between $X(t)$ and X^* with initial condition $\tilde{X}(0) = X(0) - X^* \in R^{m \times n}$, dynamic equation (10) is further derived equivalently as the following:

$$\begin{aligned} \dot{\tilde{X}}(t) &= -\gamma \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s \tilde{X}(t) B_s B_r^T + \gamma \sum_{l=1}^p A_l^T C B_l^T \\ &\quad - \gamma \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s X^* B_s B_r^T \\ &= -\gamma \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s \tilde{X}(t) B_s B_r^T \\ &\quad + \gamma \sum_{r=1}^p A_r^T \left(C - \sum_{s=1}^p A_s X^* B_s \right) B_r^T. \end{aligned} \quad (11)$$

With $\sum_{l=1}^p A_l X^* B_l = C$ considered, (11) can be simplified as

$$\dot{\tilde{X}}(t) = -\gamma \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s \tilde{X}(t) B_s B_r^T. \quad (12)$$

Similarly, we define the following Lyapunov-function candidate:

$$V(\tilde{X}(t), t)_{(10)} = \|\tilde{X}(t)\|_F^2 \geq 0, \quad (13)$$

with its time derivative being

$$\begin{aligned} \dot{V}(\tilde{X}(t), t)_{(10)} &= \text{trace} \left[\left(\frac{\partial V(\tilde{X}(t), t)}{\partial \tilde{X}(t)} \right)^T \dot{\tilde{X}}(t) \right] \\ &= \text{trace} \left(-2\gamma \tilde{X}^T(t) \sum_{s=1}^p \sum_{r=1}^p A_r^T A_s \tilde{X}(t) B_s B_r^T \right) \\ &= -2\gamma \left\| \sum_{l=1}^p A_l \tilde{X}(t) B_l \right\|_F^2 \leq 0. \end{aligned} \quad (14)$$

There exists a positive scalar $\alpha > 0$ [23] being the minimum eigenvalue of $(\sum_{l=1}^p B_l^T \otimes A_l)^T (\sum_{l=1}^p B_l^T \otimes A_l)$ satisfying

$$\left\| \sum_{l=1}^p A_l \tilde{X}(t) B_l \right\|_F^2 \geq \alpha \|\tilde{X}(t)\|_F^2 \quad (15)$$

if the unique solution condition of GLME (1) holds. Thus, we can have

$$\dot{V}(\tilde{X}(t), t) \leq -2\gamma\alpha V(\tilde{X}(t), t); \quad (16)$$

that is,

$$V(\tilde{X}(t), t) \leq \exp(-2\gamma\alpha t) V(\tilde{X}(0), 0), \quad (17)$$

which could be further equivalently rewritten as

$$\|X(t) - X^*\|_F \leq \exp(-\gamma\alpha t) \|X(0) - X^*\|_F. \quad (18)$$

By Lyapunov theory [20], (14) and (18) indicate that state matrix $X(t) \in R^{m \times n}$ of (10) can globally and exponentially converge to the unique theoretical solution $X^* \in R^{m \times n}$ of GLME (1). The proof is thus complete. \square

It is worth noting that if GLME (1) is with multiple theoretical solutions $X^* \in R^{m \times n}$, scalar α equals zero. In this situation, the linear model (10) at least could guarantee its global convergence but is not able with explicit exponential convergence rate.

3. Superior Convergence with Specific Nonlinear Activation Functions

According to Theorem 1, The odd monotonically increasing activation function is able to guarantee global convergence of the general recurrent neural network (2). If the linear activation function is adopted, the general recurrent neural network model reduces to the linear model (10). Such linear model (10) possesses global exponential convergence property. In order to achieve superior convergence to global exponential convergence of the linear model (10), specific types of nonlinear activation functions should be chosen

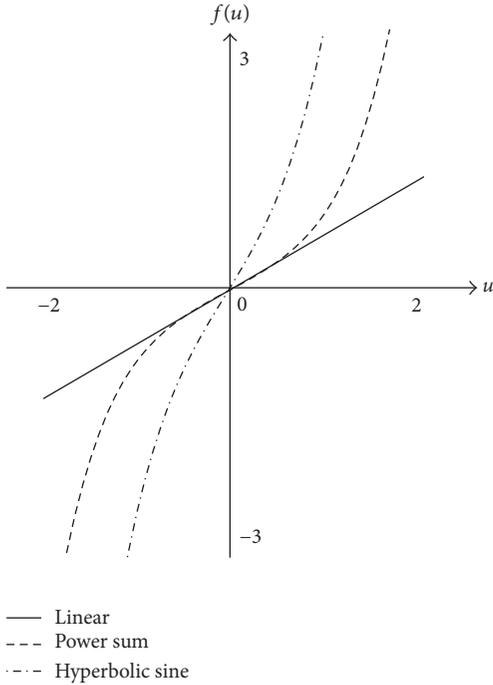


FIGURE 1: The general recurrent neural network model (2) utilizes the three types of activation functions: linear, power sum ($N = 3$), and hyperbolic sine ($\xi = 3$).

properly. Owing to the aforementioned considerations, two types of nonlinear activation functions, power sum and hyperbolic sine functions, are proposed to activate the general recurrent neural network model (2). Figure 1 shows the curve plotting of the three aforementioned activation functions used in (2). Correspondingly, we will have the following theorems on the two neural network models' convergence properties.

Theorem 3. *If the general recurrent neural network (2) is activated by the power sum function $f(u) = \sum_{k=1}^N u^{2k-1}$, the state matrix $X(t) \in \mathbb{R}^{m \times n}$ of (2) can globally and superiorly converge to the unique theoretical solution $X^* \in \mathbb{R}^{m \times n}$, as compared with linear model (10).*

Proof. In order to prove the convergence property of (2) activated by the power sum function $f(u) = \sum_{k=1}^N u^{2k-1}$ under this situation, we define the following Lyapunov-function candidate:

$$V_{\text{ps}}(\bar{X}(t), t) := V(\bar{X}(t), t)_{(3)} = V(\bar{X}(t), t)_{(10)}, \quad (19)$$

with its time derivative being

$$\begin{aligned} \dot{V}_{\text{ps}}(\bar{X}(t), t) &= -2\gamma \Delta^T \mathcal{F}(\Delta) = -2\gamma \sum_{i=1}^{mn} \Delta_i f(\Delta_i) \\ &= -2\gamma \sum_{i=1}^{mn} \Delta_i \sum_{k=1}^N (\Delta_i)^{2k-1} \end{aligned}$$

$$\begin{aligned} &= -2\gamma \sum_{i=1}^{mn} \sum_{k=1}^N (\Delta_i)^{2k} \leq -2\gamma \sum_{i=1}^{mn} (\Delta_i)^2 \\ &= \dot{V}(\bar{X}(t), t)_{(10)}, \end{aligned} \quad (20)$$

where

$$\Delta = [\Delta_1, \Delta_2, \dots, \Delta_i, \dots, \Delta_{mn}]^T \in \mathbb{R}^{mn} \quad (21)$$

and $\Delta_i \in \mathbb{R}$ denotes the i th element of vector $(\sum_{i=1}^p B_i^T \otimes A_i) \text{vec}(\bar{X}(t)) \in \mathbb{R}^{mn}$. This implies that when power sum functions are used, (2) possesses global convergence to zero matrix, with larger Lyapunov-function vanishing rate (i.e., faster convergence), as compared with the situation of (10). The proof is thus complete. \square

Theorem 4. *If the general recurrent neural network (2) is activated by hyperbolic sine function $f(u) = \exp(\xi u)/2 - \exp(-\xi u)/2$ with coefficient $\xi \geq 1$, the state matrix $X(t) \in \mathbb{R}^{m \times n}$ of (2) can globally and superiorly converge to the unique theoretical solution $X^* \in \mathbb{R}^{m \times n}$, as compared with the linear model (10).*

Proof. Similarly, the following Lyapunov function is defined to investigate convergence:

$$V_{\text{hs}}(\bar{X}(t), t) := V_{\text{ps}}(\bar{X}(t), t) = V(\bar{X}(t), t)_{(10)}, \quad (22)$$

and its time derivative is

$$\begin{aligned} \dot{V}_{\text{hs}}(\bar{X}(t), t) &= -2\gamma \sum_{i=1}^{mn} \Delta_i f(\Delta_i) \\ &= -\gamma \sum_{i=1}^{mn} \Delta_i (\exp(\xi \Delta_i) - \exp(-\xi \Delta_i)) \\ &= -2\gamma \sum_{i=1}^{mn} \sum_{j=1}^{+\infty} \frac{(\xi \Delta_i)^{2j}}{(2j-1)!} \leq -2\gamma \sum_{i=1}^{mn} (\Delta_i)^2 \\ &= \dot{V}(\bar{X}(t), t)_{(10)}, \end{aligned} \quad (23)$$

which indicates that when the hyperbolic sine activation function is employed, the nonlinear recurrent neural network model (2) possesses global convergence as its state matrix is approaching zero, with larger Lyapunov-function vanishing rate (i.e., faster convergence) as compared with the situation of linear model (10). These complete the proof. \square

4. Illustrative Examples

In this section, three examples are presented to illustrate the efficiency of the general nonlinear recurrent neural network (2) with its specific models under different types of activation functions (linear, power sum, and hyperbolic sine activation functions) for online solving GLME (1).

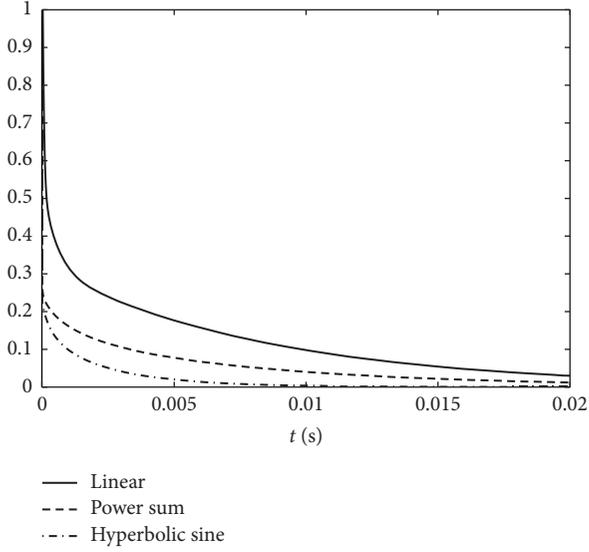


FIGURE 2: Solution error $\|X(t) - X^*\|_F$ synthesized by (2) for GLME (24) with the unique theoretical solution X^* , starting from the same randomly generated initial state $X(0) \in R^{2 \times 2}$.

Example 1. Let us consider the following GLME with $l = 2$:

$$A_1XB_1 + A_2XB_2 = C, \quad (24)$$

where

$$\begin{aligned} A_1 &= \begin{bmatrix} 2 & 1 \\ 1 & 3 \end{bmatrix}, \\ A_2 &= \begin{bmatrix} 6 & 2 \\ 3 & 4 \end{bmatrix}, \\ B_1 &= \begin{bmatrix} 5 & 3 \\ 3 & 4 \end{bmatrix}, \\ B_2 &= \begin{bmatrix} 6 & 2 \\ 5 & 8 \end{bmatrix}, \\ C &= \begin{bmatrix} 3 & 4 \\ 5 & 6 \end{bmatrix}. \end{aligned} \quad (25)$$

GLME (24) has unique theoretical solution

$$X^* = \begin{bmatrix} -0.0023 & 0.0308 \\ 0.0395 & 0.1019 \end{bmatrix}, \quad (26)$$

since the eigenvalues of coefficient matrices A_1 , A_2 , B_1 , and B_2 are all positive values. We employ the general recurrent neural network model (2) with $\gamma = 1$ activated by linear function, power sum function with $N = 4$, and hyperbolic sine function with $\xi = 3$.

From Figure 2, we could observe that the solution errors $\|X(t) - X^*\|_F$ decline to almost zero at around 0.02 s and faster convergence to the solution can be achieved with power sum

and hyperbolic sine activation functions used in (2). These can demonstrate the effectiveness of the general recurrent neural network model (2) for solving GLME (24).

Example 2. Let us consider the following GLME with multiple theoretical solutions $X^* \in R^{2 \times 2}$:

$$\tilde{A}_1X\tilde{B}_1 + \tilde{A}_2X\tilde{B}_2 = \tilde{C}, \quad (27)$$

where

$$\begin{aligned} \tilde{A}_1 &= \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \\ \tilde{A}_2 &= \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix}, \\ \tilde{B}_1 &= \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \\ \tilde{B}_2 &= \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \\ \tilde{C} &= \begin{bmatrix} 0 & 3 \\ 0 & 5 \end{bmatrix}. \end{aligned} \quad (28)$$

We use linear model (10) with design parameter $\gamma = 1$ to solve GLME (27). The trajectories of entries of state matrix $X(t) \in R^{2 \times 2}$ are shown in Figure 3. From Figure 3, we could see that, starting from two different initial matrices $X(0) \in R^{2 \times 2}$, the state matrices $X(t) \in R^{2 \times 2}$ of linear model (10), respectively, converge to two different trajectories (or say two different theoretical solutions $X^* \in R^{2 \times 2}$). This indicates that the choices of the initial value impact greatly the steady-state results of the recurrent neural network (2) and determine the starting points of convergence for solution of GLME (27), if multiple theoretical solutions exist for GLME (27). Correspondingly, the residual errors $\|\tilde{A}_1X(t)\tilde{B}_1 + \tilde{A}_2X(t)\tilde{B}_2 - \tilde{C}\|_F$ synthesized by (10) can always diminish to zero within finite time from twenty different initial values, as illustrated by Figure 4.

Example 3. Let us consider the following GLME in a larger dimension with $l = 10$:

$$\sum_{l=1}^{10} A_lXB_l = C, \quad (29)$$

where coefficient matrices $A_l \in R^{10 \times 10}$, $B_l \in R^{10 \times 10}$, and $C \in R^{10 \times 10}$ are all positive-definite randomly generated and fall within interval $[-2, 2] \in R^{10 \times 10}$. We exploit nonlinear neural network models (2) activated by power sum and hyperbolic sine functions and the linear model (10) to solve GLME (29) with design parameter $\gamma = 1$. From Table 1, we could observe that general recurrent neural network models (2) activated by power sum and hyperbolic sine activation functions exhibit faster error diminishing speed than that of the linear model (10), with all of their residual errors reaching the level of

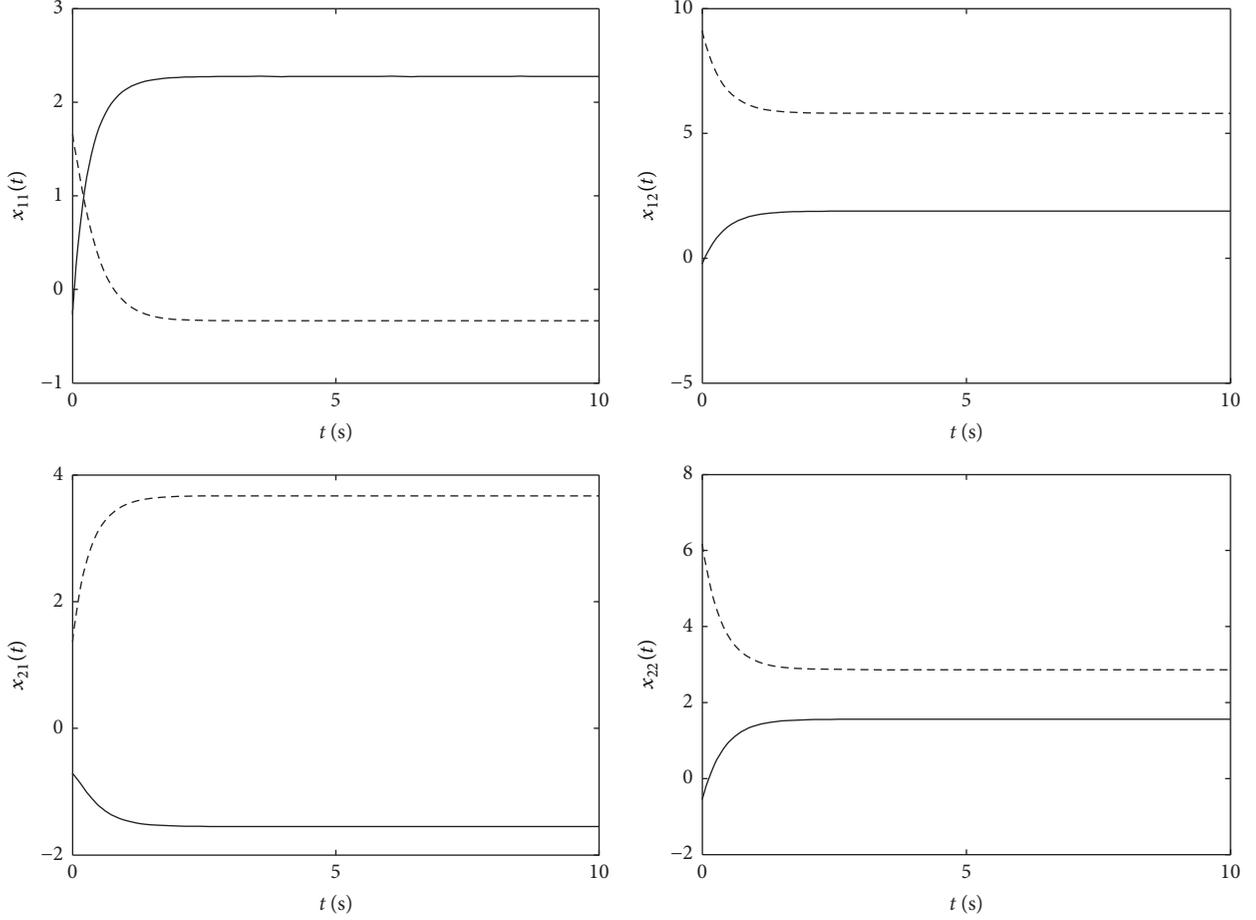


FIGURE 3: Trajectories of state matrix $X(t) \in \mathbb{R}^{2 \times 2}$ for solving GLME (27) with the multiple theoretical solutions X^* , starting from two different initial states $X(0) \in \mathbb{R}^{2 \times 2}$.

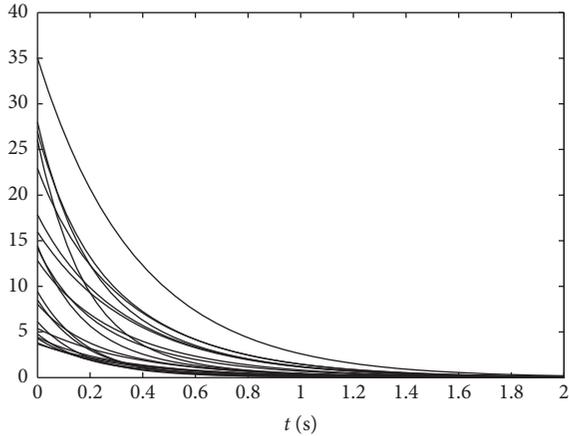


FIGURE 4: Residual error $\|\tilde{A}_1 X(t) \tilde{B}_1 + \tilde{A}_2 X(t) \tilde{B}_2 - \tilde{C}\|_F$ synthesized by (2) for solution of GLME (27), starting from twenty different initial states $X(0) \in \mathbb{R}^{2 \times 2}$.

10^{-7} within 1s. From computational results of these three examples above, we could see that the proposed general nonlinear recurrent neural network can solve the GLME (1) problem well.

TABLE 1: Performance of the general recurrent neural network model (2) with three different activation functions (linear, power sum, and hyperbolic sine) for solving GLME (29).

Time t (s)	Residual error $\ \sum_{l=1}^{10} A_l X(t) B_l - C\ _F$		
	Linear	Power sum	Hyperbolic sine
$t = 0$ s	37.4543	37.4543	37.4543
$t = 0.1$ s	0.9886	0.3678	0.5242
$t = 1$ s	4.6368×10^{-7}	9.7531×10^{-8}	3.0543×10^{-7}

5. Conclusion

In this brief, we present a general recurrent neural network model for solving GLME. The general nonlinear model of recurrent neural network possesses global convergence property in finding solutions of GLME. By using specifically proposed nonlinear activation functions, superior convergence can be achieved, as compared with the linear model which is with exponential convergence rate. Illustrative results are shown to demonstrate the effectiveness and superiority of nonlinear recurrent neural network models for solution of GLME.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (NSFC) under Grant no. 61603078 and the Fundamental Research Funds for the Central Universities at the University of Electronic Science and Technology of China (UESTC) under Grant no. ZYGX2015KYQD044.

References

- [1] V. Hernandez, E. S. Quintana, and M. Marques, "Solving linear matrix equations in control problems on distributed memory multiprocessors," in *Proceedings of the 33rd IEEE Conference on Decision and Control. Part 1 (of 4)*, pp. 448–454, December 1994.
- [2] H. Lev-Ari, "Efficient solution of linear matrix equations with application to multistatic antenna array processing," *Communications in Information & Systems*, vol. 5, no. 1, pp. 123–130, 2005.
- [3] Y. Fang, K. A. Loparo, and X. Feng, "New estimates for solutions of Lyapunov equations," *Institute of Electrical and Electronics Engineers. Transactions on Automatic Control*, vol. 42, no. 3, pp. 408–411, 1997.
- [4] W. H. Kwon, Y. S. Moon, and S. C. Ahn, "Bounds in algebraic Riccati and Lyapunov equations: a survey and some new results," *International Journal of Control*, vol. 64, no. 3, pp. 377–389, 1996.
- [5] B. Zhou and G.-R. Duan, "A new solution to the generalized Sylvester matrix equation $av-evf= bw$," *Systems & Control Letters*, vol. 2, no. 3, pp. 193–198, 2006.
- [6] B. Zhou and G.-R. Duan, "On the generalized Sylvester mapping and matrix equations," *Systems & Control Letters*, vol. 57, no. 3, pp. 200–208, 2008.
- [7] S. Li and Y. Li, "Nonlinearly activated neural network for solving time-varying complex Sylvester equation," *IEEE Transactions on Cybernetics*, vol. 44, no. 8, pp. 1397–1407, 2014.
- [8] Y. Zhang, D. Jiang, and J. Wang, "A recurrent neural network for solving Sylvester equation with time-varying coefficients," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1053–1063, 2002.
- [9] F. Ding and T. Chen, "Gradient based iterative algorithms for solving a class of matrix equations," *IEEE Transactions on Automatic Control*, vol. 50, no. 8, pp. 1216–1221, 2005.
- [10] L. Xie, J. Ding, and F. Ding, "Gradient based iterative solutions for general linear matrix equations," *Computers & Mathematics with Applications*, vol. 58, no. 7, pp. 1441–1448, 2009.
- [11] F. Ding and T. Chen, "Iterative least-squares solutions of coupled Sylvester matrix equations," *Systems & Control Letters*, vol. 54, no. 2, pp. 95–107, 2005.
- [12] Y. Chen, C. Yi, and D. Qiao, "Improved neural solution for the Lyapunov matrix equation based on gradient search," *Information Processing Letters*, vol. 113, no. 22–24, pp. 876–881, 2013.
- [13] K. Chen, "Improved neural dynamics for online Sylvester equations solving," *Information Processing Letters*, vol. 116, no. 7, pp. 455–459, 2016.
- [14] S. Zhang and A. G. Constantinides, "Lagrange programming neural networks," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 39, no. 7, pp. 441–452, 1992.
- [15] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 79, no. 8, pp. 2554–2558, 1982.
- [16] J. Wang, "Recurrent neural networks for solving linear matrix equations," *Computers & Mathematics with Applications*, vol. 26, no. 9, pp. 23–34, 1993.
- [17] Z. Zhang and Y. Zhang, "Design and experimentation of acceleration-level drift-free scheme aided by two recurrent neural networks," *IET Control Theory Applications*, vol. 7, no. 1, pp. 25–42, 2013.
- [18] Z. Zhang, Z. Li, Y. Zhang, Y. Luo, and Y. Li, "Neural-dynamic-method-based dual-arm CMG scheme with time-varying constraints applied to humanoid robots," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3251–3262, 2015.
- [19] C. Mead and M. Ismail, *Analog VLSI Implementation of Neural Systems*, vol. 80, Springer Science & Business Media, 2012.
- [20] S. Sastry and M. Bodson, *Adaptive Control: Stability, Convergence and Robustness*, Courier Corporation, 2011.
- [21] J. Liu, M. A. Brooke, and K. Hirotsu, "A CMOS feedforward neural-network chip with on-chip parallel learning for oscillation cancellation," *IEEE Transactions on Neural Networks*, vol. 13, no. 5, pp. 1178–1186, 2002.
- [22] A. H. Roger and R. J. Charles, *Matrix Analysis*, Cambridge University Press, New York, NY, USA, 2nd edition, 2012.
- [23] Y. Zhang and K. Chen, "Global exponential convergence and stability of Wang neural network for solving online linear equations," *Electronics Letters*, vol. 44, no. 2, pp. 145–146, 2008.

Research Article

Minimal-Learning-Parameter Technique Based Adaptive Neural Sliding Mode Control of MEMS Gyroscope

Bin Xu and Pengchao Zhang

Shaanxi Provincial Key Laboratory of Industrial Automation, Shaanxi University of Technology, Hanzhong, Shaanxi 723000, China

Correspondence should be addressed to Bin Xu; smileface.binxu@gmail.com

Received 15 May 2017; Accepted 27 June 2017; Published 26 July 2017

Academic Editor: Yanan Li

Copyright © 2017 Bin Xu and Pengchao Zhang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper investigates an adaptive neural sliding mode controller for MEMS gyroscopes with minimal-learning-parameter technique. Considering the system uncertainty in dynamics, neural network is employed for approximation. Minimal-learning-parameter technique is constructed to decrease the number of update parameters, and in this way the computation burden is greatly reduced. Sliding mode control is designed to cancel the effect of time-varying disturbance. The closed-loop stability analysis is established via Lyapunov approach. Simulation results are presented to demonstrate the effectiveness of the method.

1. Introduction

Recently, MEMS gyroscopes have been drawing growing attention because they intend to employ advanced control approaches to realize trajectories tracking and to handle system parametric uncertainties and disturbances. These intelligent control methods improve the performance of gyroscopes, so that the applications of MEMS gyroscopes are expanded. With the vigorous development of nonlinear system control methods [1–8], a variety of gyroscopic modal control methods emerged.

In [9, 10], one adaptive operation strategy is presented to control the MEMS z -axis gyroscope, which offers a larger operational bandwidth, absence of zero-rate output, self-calibration, and large robustness to parameter variations caused by fabrication defects and ambient conditions. In [11, 12], the sliding mode control is proposed to handle the vibrating proof mass, which achieves better estimation of the unknown angular velocity than conventional model reference adaptive feedback controller. Since then, in the presence of significant uncertainties, the regulated model-based and non-model-based sliding model control approaches are presented to improve tracking control of the drive and sense modes of the vibratory gyroscope in [13]. In [14], an adaptive tracking

controller with a proportional and integral sliding surface is proposed.

Neural network has an inherent ability to learn and approximate nonlinear functions [15, 16], which can be utilized for unstructured uncertainties. Thus, an adaptive control strategy using radial basis function (RBF) network/Fuzzy Logic System compensator is presented for robust tracking of MEMS gyroscope in the presence of model uncertainties and external disturbances to compensate such system nonlinearities and improve the tracking performance in [17–19]. However, in practical application, large amount of update parameters results in the computation burden of online learning. In [20], the minimal-learning-parameter technique is further incorporated into the high gain observer to greatly reduce the online computation burden.

Inspired by the above-mentioned discussions on designing intelligent controllers and reducing the number of online parameters, this paper will focus on constructing the new control scheme for MEMS gyroscopes to suppress the system parametric uncertainties and disturbances. The main contribution of this paper is that a single parameter is employed to replace weight matrix, which significantly reduces the computation burden.

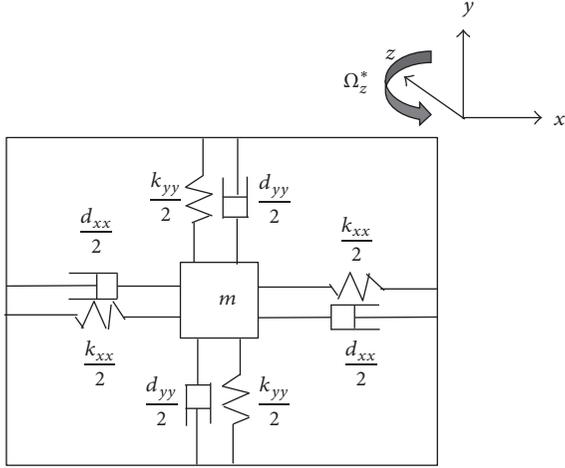


FIGURE 1: The basic principle diagram of z-axis MEMS gyroscope.

The structure of this paper is organized as follows. Section 2 formulates the dynamics of MEMS gyroscope. Section 3 studies the neural network of lumped parametric uncertainties. In Section 4, an adaptive neural sliding mode control strategy using the minimal-learning-parameter technique is designed and stability analysis is discussed. Numerical simulations are investigated to verify the superiority of the proposed approach in Section 5. Conclusions are given in Section 6.

2. Problem Formulation

2.1. Dynamics of MEMS Gyroscope. As Figure 1 shows, the ideal MEMS gyroscope is a quality-stiffness-damping system. Considering the mechanical coupling caused by manufacturing defects, the dynamics of MEMS gyroscopes can be expressed as

$$\begin{aligned} m\ddot{x} + d_{xx}\dot{x} + (d_{xy} - 2m\Omega_z^*)\dot{y} + (k_{xx} - m\Omega_z^{*2})x \\ + k_{xy}y = u_x^*, \\ m\ddot{y} + d_{yy}\dot{y} + (d_{xy} + 2m\Omega_z^*)\dot{x} + (k_{yy} - m\Omega_z^{*2})y \\ + k_{xy}x = u_y^*, \end{aligned} \quad (1)$$

where m represents the mass of proof mass, Ω_z^* represents the input angular velocity, x and y represent the system generalized coordinates, d_{xx} and d_{yy} represent damping terms, d_{xy} represents asymmetric damping term, k_{xx} and k_{yy} represent spring terms, k_{xy} represents asymmetric spring terms, and u_x^* and u_y^* represent the control forces.

In (1), the damping terms are affected by atmospheric pressure and spring terms are affected by ambient temperature. That means

$$\begin{aligned} d_{xx} &= d_{x0} + \Delta d_{xx}, \\ d_{yy} &= d_{y0} + \Delta d_{yy}, \end{aligned}$$

$$k_{xx} = k_{x0} + \Delta k_{xx},$$

$$k_{yy} = k_{y0} + \Delta k_{yy},$$

$$k_{xy} = k_{xy0} + \Delta k_{xy},$$

(2)

where d_{x0} , d_{y0} , and d_{xy0} are the damping terms of MEMS gyroscope in normal atmospheric pressure, k_{x0} , k_{y0} , and k_{xy0} are the spring terms of MEMS gyroscope in room temperature environment, Δd_{xx} , Δd_{yy} , and Δd_{xy} are the deviation in damping coefficients due to the changes of atmospheric pressure, and Δk_{xx} , Δk_{yy} , and Δk_{xy} are the deviation of damping coefficients because of the changes of ambient temperature.

Dividing both sides of (1) by reference mass m , reference frequency ω_o^2 , and reference length q_o , the dynamics can be derived as

$$\begin{aligned} \frac{\mathbf{q}^*}{q_o} + \frac{\mathbf{D}^*}{m\omega_o} \frac{\mathbf{q}^*}{q_o} + 2\frac{\mathbf{S}^*}{\omega_o} \frac{\mathbf{q}^*}{q_o} - \frac{\Omega_z^{*2}}{\omega_o^2} \frac{\mathbf{q}^*}{q_o} + \frac{\mathbf{K}^*}{m\omega_o^2} \frac{\mathbf{q}^*}{q_o} \\ = \frac{\mathbf{u}^*}{m\omega_o^2 q_o}, \end{aligned} \quad (3)$$

where $\mathbf{q}^* = \begin{bmatrix} x \\ y \end{bmatrix}$, $\mathbf{u}^* = \begin{bmatrix} u_x^* \\ u_y^* \end{bmatrix}$, $\mathbf{D}^* = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{xy} & d_{yy} \end{bmatrix}$, $\mathbf{S}^* = \begin{bmatrix} 0 & -\Omega_z^* \\ \Omega_z^* & 0 \end{bmatrix}$, and $\mathbf{K}^* = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{xy} & k_{yy} \end{bmatrix}$.

Define new parameters as $\mathbf{q} = \mathbf{q}^*/q_o$, $\mathbf{u} = \mathbf{u}^*/m\omega_o^2 q_o$, $\Omega_z = \Omega_z^*/\omega_o$, $\mathbf{D} = \mathbf{D}^*/m\omega_o$, $\mathbf{K} = \mathbf{K}^*/m\omega_o^2$, and $\mathbf{S} = -\mathbf{S}^*/\omega_o$.

Then, (3) has the following form:

$$\ddot{\mathbf{q}} = (2\mathbf{S} - \mathbf{D})\dot{\mathbf{q}} + (\Omega_z^2 - \mathbf{K})\mathbf{q} + \mathbf{u}, \quad (4)$$

where $\mathbf{D} = \mathbf{D}_0 + \Delta\mathbf{D}$, $\mathbf{K} = \mathbf{K}_0 + \Delta\mathbf{K}$ and $\mathbf{D}_0 = \begin{bmatrix} d_{x0}/m\omega_o & d_{xy0}/m\omega_o \\ d_{xy0}/m\omega_o & d_{y0}/m\omega_o \end{bmatrix}$, $\mathbf{K}_0 = \begin{bmatrix} k_{x0}/m\omega_o^2 & k_{xy0}/m\omega_o^2 \\ k_{xy0}/m\omega_o^2 & k_{y0}/m\omega_o^2 \end{bmatrix}$, $\Delta\mathbf{D} = \begin{bmatrix} \Delta d_{xx}/m\omega_o & \Delta d_{xy}/m\omega_o \\ \Delta d_{xy}/m\omega_o & \Delta d_{yy}/m\omega_o \end{bmatrix}$, and $\Delta\mathbf{K} = \begin{bmatrix} \Delta k_{xx}/m\omega_o^2 & \Delta k_{xy}/m\omega_o^2 \\ \Delta k_{xy}/m\omega_o^2 & \Delta k_{yy}/m\omega_o^2 \end{bmatrix}$.

With the external disturbances caused by compound maneuvers, (4) is replaced by

$$\begin{aligned} \ddot{\mathbf{q}} = (2\mathbf{S} - \mathbf{D}_0 - \Delta\mathbf{D})\dot{\mathbf{q}} + (\Omega_z^2 - \mathbf{K}_0 - \Delta\mathbf{K})\mathbf{q} + \mathbf{Cu} \\ + \mathbf{d}(\mathbf{t}), \end{aligned} \quad (5)$$

where $\mathbf{d}(\mathbf{t})$ is the external disturbance and $\mathbf{d}_m = \sup|\mathbf{d}(\mathbf{t})|$.

Define lumped parametric uncertainties as

$$\mathbf{P}(\dot{\mathbf{q}}, \mathbf{q}) = -\Delta\mathbf{D}\dot{\mathbf{q}} - \Delta\mathbf{K}\mathbf{q}. \quad (6)$$

Equation (5) can be written as

$$\ddot{\mathbf{q}} = \mathbf{A}\dot{\mathbf{q}} + \mathbf{B}\mathbf{q} + \mathbf{Cu} + \mathbf{P}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{t}), \quad (7)$$

where $\mathbf{A} = 2\mathbf{S} - \mathbf{D}_0$ and $\mathbf{B} = \Omega_z^2 - \mathbf{K}_0$.

Remark 1. In order to make the uncertainty of MEMS gyroscope depicted in (6) controllable, there must be unknown matrices of appropriate dimensions \mathbf{G} , \mathbf{H} , and $\mathbf{p}(\mathbf{t})$ such that $\Delta\mathbf{A} = \mathbf{C}\mathbf{G}$, $\Delta\mathbf{B} = \mathbf{C}\mathbf{H}$, and $\mathbf{d}(\mathbf{t}) = \mathbf{C}\mathbf{p}(\mathbf{t})$. And \mathbf{C} is selected as $\mathbf{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$.

2.2. Control Goal. The control goal of this paper is to design a controller to steer the position q and the speed \dot{q} to the desired trajectories $\mathbf{q}_m(t) = [x_m \ y_m]^T$ and $\dot{\mathbf{q}}_m(t) = [\dot{x}_m \ \dot{y}_m]^T$. Besides, the minimal-learning-parameter technique is further incorporated into the estimation of lumped parametric uncertainties $\mathbf{P}(\dot{\mathbf{q}}, \mathbf{q})$.

3. Brief Description of RBF Neural Network

A neural network is established to approximate the lumped parametric uncertainties $\mathbf{P}(\dot{\mathbf{q}}, \mathbf{q})$, which can be expressed as

$$\hat{\mathbf{P}}(\dot{\mathbf{q}}, \mathbf{q} | \boldsymbol{\theta}) = \hat{\boldsymbol{\theta}}^T \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}), \quad (8)$$

where $\hat{\boldsymbol{\theta}} \subseteq \mathbf{R}^n$ is the adjustable parameter matrix, $\boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q})$ is a nonlinear vector function of the inputs, and the RBF has the form

$$\mu_i = \exp\left(-\frac{\|\mathbf{q} - \mathbf{q}_{mi}\|}{2\sigma_i^2}\right), \quad i = 1, 2, \dots, n, \quad (9)$$

where \mathbf{q}_{mi} is an n -dimensional vector representing the center of the i th basis function and σ_i is the variance representing the spread of the basis function.

Suppose that $\boldsymbol{\theta}^*$ are the optimal weight parameters; parametric uncertainties could be reexpressed as

$$\mathbf{P}(\dot{\mathbf{q}}, \mathbf{q}) = \boldsymbol{\theta}^{*T} \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}) + \boldsymbol{\varepsilon}, \quad (10)$$

where $\boldsymbol{\varepsilon}$ is the optimal estimation error of RBF neural network and $\boldsymbol{\varepsilon}_n = \sup|\boldsymbol{\varepsilon}|$.

Thus, the estimation error can be written as

$$\begin{aligned} \mathbf{P} - \hat{\mathbf{P}} &= \boldsymbol{\theta}^{*T} \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}) + \boldsymbol{\varepsilon} - \hat{\boldsymbol{\theta}}^T \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}) \\ &= -\tilde{\boldsymbol{\theta}}^T \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}) + \boldsymbol{\varepsilon}, \end{aligned} \quad (11)$$

where $\tilde{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^*$.

4. Adaptive Sliding Mode Control with Minimal-Learning-Parameter Technique

Define the system tracking error as

$$\mathbf{e}(\mathbf{t}) = \mathbf{q}(\mathbf{t}) - \mathbf{q}_m(\mathbf{t}). \quad (12)$$

Select the sliding mode function as

$$\mathbf{s}(\mathbf{t}) = \dot{\mathbf{e}}(\mathbf{t}) + \boldsymbol{\beta}\mathbf{e}(\mathbf{t}), \quad (13)$$

where $\boldsymbol{\beta}$ is satisfied with Hurwitz condition.

The derivative of $\mathbf{s}(\mathbf{t})$ is

$$\begin{aligned} \dot{\mathbf{s}}(\mathbf{t}) &= \ddot{\mathbf{e}}(\mathbf{t}) + \boldsymbol{\beta}\dot{\mathbf{e}}(\mathbf{t}) = [\ddot{\mathbf{q}}(\mathbf{t}) - \ddot{\mathbf{q}}_m(\mathbf{t})] + \boldsymbol{\beta}\dot{\mathbf{e}}(\mathbf{t}) \\ &= \mathbf{A}\dot{\mathbf{q}} + \mathbf{B}\mathbf{q} + \mathbf{C}\mathbf{u} + \mathbf{P}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{t}) - \ddot{\mathbf{q}}_m(\mathbf{t}) \\ &\quad + \boldsymbol{\beta}\dot{\mathbf{e}}(\mathbf{t}). \end{aligned} \quad (14)$$

Define $\phi = \|\boldsymbol{\theta}^*\|^2$ and the estimation error is $\tilde{\phi} = \hat{\phi} - \phi$, where $\hat{\phi}$ is the estimation of ϕ .

Assume that $\dot{\mathbf{s}} = \mathbf{0}$; according to (14), controller could be designed as

$$\begin{aligned} \mathbf{u} &= \mathbf{C}^{-1} \left[\ddot{\mathbf{q}}_m - \mathbf{A}\dot{\mathbf{q}} - \mathbf{B}\mathbf{q} - \frac{1}{2}\mathbf{s}\hat{\phi}\boldsymbol{\mu}^T\boldsymbol{\mu} - \hat{\boldsymbol{\eta}}_m * \text{sgn}(\mathbf{s}) \right. \\ &\quad \left. - \boldsymbol{\beta}\dot{\mathbf{e}} - \mathbf{K}\mathbf{s} \right], \end{aligned} \quad (15)$$

where $\hat{\boldsymbol{\eta}}_m = \sup|\boldsymbol{\eta}|$, $\boldsymbol{\eta} = \boldsymbol{\varepsilon} + \mathbf{d}(\mathbf{t})$, $\hat{\boldsymbol{\eta}}_m * \text{sgn}(\mathbf{s})$ is Hadamard product item, and $-\mathbf{C}^{-1}\mathbf{K}\mathbf{s}$ is a robust item.

Remark 2. Compared with traditional results of MEMS control [15–17], in this paper, minimal-learning-parameter technique is employed for controller design to reduce computation burden.

Remark 3. When strong time-varying disturbances exist, the boundary layer is bigger than before and the estimation errors are increased.

The adaptive law of single parameter $\hat{\phi}$ can be designed as

$$\dot{\hat{\phi}} = \frac{\gamma}{2}\mathbf{s}^T\mathbf{s}\boldsymbol{\mu}^T\boldsymbol{\mu} - \kappa\gamma\hat{\phi}, \quad (16)$$

where $\gamma > 0$ and $\kappa > 0$.

An adaptive item $\hat{\boldsymbol{\eta}}_m$ is employed to estimate $\boldsymbol{\eta}_m$, and the estimated error is $\tilde{\boldsymbol{\eta}}_m = \hat{\boldsymbol{\eta}}_m - \boldsymbol{\eta}_m$. The adaptive law of $\hat{\boldsymbol{\eta}}_m$ is selected as

$$\dot{\hat{\boldsymbol{\eta}}}_m = \tau(|\mathbf{s}| - \alpha\hat{\boldsymbol{\eta}}_m), \quad (17)$$

where $\tau > 0$ and $\alpha > 0$.

Substitute (15) into (14):

$$\begin{aligned} \dot{\mathbf{s}}(\mathbf{t}) &= \left[\ddot{\mathbf{q}}_m - \mathbf{A}\dot{\mathbf{q}} - \mathbf{B}\mathbf{q} - \frac{1}{2}\mathbf{s}\hat{\phi}\boldsymbol{\mu}^T\boldsymbol{\mu} - \hat{\boldsymbol{\eta}}_m * \text{sgn}(\mathbf{s}) \right. \\ &\quad \left. - \boldsymbol{\beta}\dot{\mathbf{e}} - \mathbf{K}\mathbf{s} \right] + \mathbf{A}\dot{\mathbf{q}} + \mathbf{B}\mathbf{q} + \mathbf{P}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{d}(\mathbf{t}) - \ddot{\mathbf{q}}_m(\mathbf{t}) \\ &\quad + \boldsymbol{\beta}\dot{\mathbf{e}}(\mathbf{t}) = -\frac{1}{2}\mathbf{s}\hat{\phi}\boldsymbol{\mu}^T\boldsymbol{\mu} - \hat{\boldsymbol{\eta}}_m * \text{sgn}(\mathbf{s}) + \left[\boldsymbol{\theta}^{*T} \boldsymbol{\mu}(\dot{\mathbf{q}}, \mathbf{q}) \right. \\ &\quad \left. + \boldsymbol{\varepsilon} \right] + \mathbf{d}(\mathbf{t}) - \mathbf{K}\mathbf{s}. \end{aligned} \quad (18)$$

Theorem 4. Considering that the nonlinear system (7) is with parametric uncertainties and disturbances, if controller (15) and updating laws (16) and (17) are designed, then the boundedness of all the closed-loop system signals included in (19) can be guaranteed.

Proof. Lyapunov function is selected as

$$L = \frac{1}{2} \mathbf{s}^T \mathbf{s} + \frac{1}{2\gamma} \tilde{\phi}^2 + \frac{1}{2\tau} \tilde{\boldsymbol{\eta}}_m^T \tilde{\boldsymbol{\eta}}_m. \quad (19)$$

The derivative of Lyapunov function is

$$\begin{aligned} \dot{L} &= \mathbf{s}^T \dot{\mathbf{s}} + \frac{1}{\gamma} \tilde{\phi} \dot{\tilde{\phi}} + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m = \mathbf{s}^T \left[-\frac{1}{2} \mathbf{s} \hat{\boldsymbol{\phi}} \boldsymbol{\mu}^T \boldsymbol{\mu} - \hat{\boldsymbol{\eta}}_m \right. \\ &\quad * \operatorname{sgn}(\mathbf{s}) + \left[\boldsymbol{\theta}^{*T} \boldsymbol{\mu}(\hat{\mathbf{q}}, \mathbf{q}) + \boldsymbol{\varepsilon} \right] + \mathbf{d}(t) - \mathbf{K}\mathbf{s} \left. \right] + \frac{1}{\gamma} \tilde{\phi} \dot{\tilde{\phi}} \\ &\quad + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m = -\frac{1}{2} \mathbf{s}^T \mathbf{s} \hat{\boldsymbol{\phi}} \boldsymbol{\mu}^T \boldsymbol{\mu} + \mathbf{s}^T \boldsymbol{\theta}^{*T} \boldsymbol{\mu}(\hat{\mathbf{q}}, \mathbf{q}) + \mathbf{s}^T [\boldsymbol{\varepsilon} \\ &\quad + \mathbf{d}(t) - \hat{\boldsymbol{\eta}}_m * \operatorname{sgn}(\mathbf{s})] - \mathbf{s}^T \mathbf{K}\mathbf{s} + \frac{1}{\gamma} \tilde{\phi} \dot{\tilde{\phi}} + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m \\ &\leq -\frac{1}{2} \mathbf{s}^T \mathbf{s} \hat{\boldsymbol{\phi}} \boldsymbol{\mu}^T \boldsymbol{\mu} + \frac{1}{2} \mathbf{s}^T \mathbf{s} \hat{\boldsymbol{\phi}} \boldsymbol{\mu}^T \boldsymbol{\mu} + \frac{1}{2} + \mathbf{s}^T [\boldsymbol{\eta} - \hat{\boldsymbol{\eta}}_m \\ &\quad * \operatorname{sgn}(\mathbf{s})] - \mathbf{s}^T \mathbf{K}\mathbf{s} + \frac{1}{\gamma} \tilde{\phi} \dot{\tilde{\phi}} + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m \leq -\frac{1}{2} \\ &\quad \cdot \mathbf{s}^T \mathbf{s} \hat{\boldsymbol{\phi}} \boldsymbol{\mu}^T \boldsymbol{\mu} + \frac{1}{2} + \mathbf{s}^T [\boldsymbol{\eta} - \boldsymbol{\eta}_m * \operatorname{sgn}(\mathbf{s}) + \boldsymbol{\eta}_m \\ &\quad * \operatorname{sgn}(\mathbf{s}) - \hat{\boldsymbol{\eta}}_m * \operatorname{sgn}(\mathbf{s})] - \mathbf{s}^T \mathbf{K}\mathbf{s} + \frac{1}{\gamma} \tilde{\phi} \dot{\tilde{\phi}} + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m \\ &\leq \tilde{\phi} \left(-\frac{1}{2} \mathbf{s}^T \mathbf{s} \boldsymbol{\mu}^T \boldsymbol{\mu} + \frac{1}{2} \dot{\tilde{\phi}} \right) + \frac{1}{2} + \mathbf{s}^T [(\boldsymbol{\eta}_m - \hat{\boldsymbol{\eta}}_m) \\ &\quad * \operatorname{sgn}(\mathbf{s})] - \mathbf{s}^T \mathbf{K}\mathbf{s} + \frac{1}{\tau} \tilde{\boldsymbol{\eta}}_m^T \dot{\tilde{\boldsymbol{\eta}}}_m. \end{aligned} \quad (20)$$

Substituting (16) and (17) into (20), the following inequality is obtained:

$$\begin{aligned} \dot{L} &\leq -\kappa \tilde{\phi} \dot{\tilde{\phi}} + \frac{1}{2} - \mathbf{s}^T \mathbf{K}\mathbf{s} - \mathbf{s}^T [\tilde{\boldsymbol{\eta}}_m * \operatorname{sgn}(\mathbf{s})] + \tilde{\boldsymbol{\eta}}_m^T |\mathbf{s}| \\ &\quad - \alpha \tilde{\boldsymbol{\eta}}_m^T \hat{\boldsymbol{\eta}}_m \leq -\frac{\kappa}{2} (\tilde{\phi}^2 - \phi^2) + \frac{1}{2} - \mathbf{s}^T \mathbf{K}\mathbf{s} - \alpha \tilde{\boldsymbol{\eta}}_m^T \hat{\boldsymbol{\eta}}_m \quad (21) \\ &= -\frac{\kappa}{2} \tilde{\phi}^2 - \mathbf{s}^T \mathbf{K}\mathbf{s} + \left(\frac{\kappa}{2} \phi^2 + \frac{1}{2} \right) - \alpha \tilde{\boldsymbol{\eta}}_m^T \hat{\boldsymbol{\eta}}_m, \end{aligned}$$

where $\kappa = 2\lambda/\gamma$ and $\lambda = \min\{\lambda_1, \lambda_2\}$, λ_1, λ_2 are the eigenvalues of matrix \mathbf{K} . Furthermore, we have

$$\begin{aligned} \dot{L} &\leq -\frac{\kappa}{2} \tilde{\phi}^2 - \mathbf{s}^T \mathbf{K}\mathbf{s} + \left(\frac{\kappa}{2} \phi^2 + \frac{1}{2} \right) \\ &\leq -\frac{\lambda}{\gamma} \tilde{\phi}^2 - \mathbf{s}^T \mathbf{K}\mathbf{s} + \left(\frac{\kappa}{2} \phi^2 + \frac{1}{2} \right) \end{aligned}$$

$$\begin{aligned} &= -2\lambda \left(\frac{1}{2\gamma} \tilde{\phi}^2 + \frac{1}{2} \mathbf{s}^T \mathbf{s} \right) + \left(\frac{\kappa}{2} \phi^2 + \frac{1}{2} \right) \\ &= -2\lambda \left(L - \frac{1}{2\tau} \tilde{\boldsymbol{\eta}}_m^T \tilde{\boldsymbol{\eta}}_m \right) + \left(\frac{\kappa}{2} \phi^2 + \frac{1}{2} \right) = -2\lambda L + Q, \end{aligned} \quad (22)$$

where $Q = (\lambda/\tau) \tilde{\boldsymbol{\eta}}_m^T \tilde{\boldsymbol{\eta}}_m + (\kappa/2) \phi^2 + 1/2$.

The solution of (22) is

$$L \leq \frac{Q}{2\lambda} + \left(L(0) - \frac{Q}{2\lambda} \right) e^{-2\lambda t}. \quad (23)$$

Then all the signals included in the Lyapunov function are bounded. This concludes the proof. \square

Remark 5. In practical application, the high-frequency switching control signals of MEMS gyroscopes result in serious chattering. Therefore, the saturation function $\operatorname{sat}(s)$ is used to replace the sign function $\operatorname{sgn}(s)$ in (15). The saturation function $\operatorname{sat}(x)$ has the form

$$\operatorname{sat}(x) = \begin{cases} 1 & x > a \\ \frac{x}{a} & |x| \leq a \\ -1 & x < -a, \end{cases} \quad (24)$$

where a is a positive constant.

5. Numerical Simulation

In this section, the aforementioned control scheme of MEMS gyroscope is simulated, the controller of which is designed as (15), and the adaptive laws are proposed as (16) and (17).

Parameters of the MEMS gyroscope are as follows:

$$\begin{aligned} m &= 0.57 \times 10^{-8} \text{ kg}, \\ d_{xx} &= 0.429 \times 10^{-6} \text{ Ns/m}, \\ d_{yy} &= 0.0429 \times 10^{-6} \text{ Ns/m}, \\ d_{xy} &= 0.0429 \times 10^{-6} \text{ Ns/m}, \\ k_{xx} &= 80.98 \text{ N/m}, \\ k_{yy} &= 71.62 \text{ N/m}, \\ k_{xy} &= 5 \text{ N/m}, \\ \Omega_z &= 5.0 \text{ rad/s}. \end{aligned} \quad (25)$$

Since the position of proof mass ranges within the scope of submillimeter and the natural frequency is generally in the range of kilohertz, the reference length is assumed as $q_o = 10 \times 10^{-6}$ m and reference frequency is assumed as $\omega_o = 1$ kHz.

Suppose that the reference trajectories are $x_m = \sin(1.71t)$, $\dot{x}_m = 1.71 \cos(1.71t)$, $y_m = 1.2 \sin(1.11t)$, and $\dot{y}_m = 1.2 \times 1.11 \cos(1.11t)$, respectively.

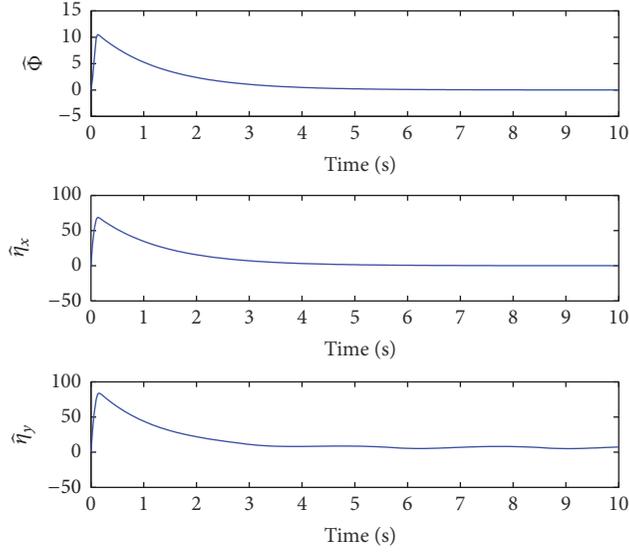


FIGURE 2: Adaptive signals.

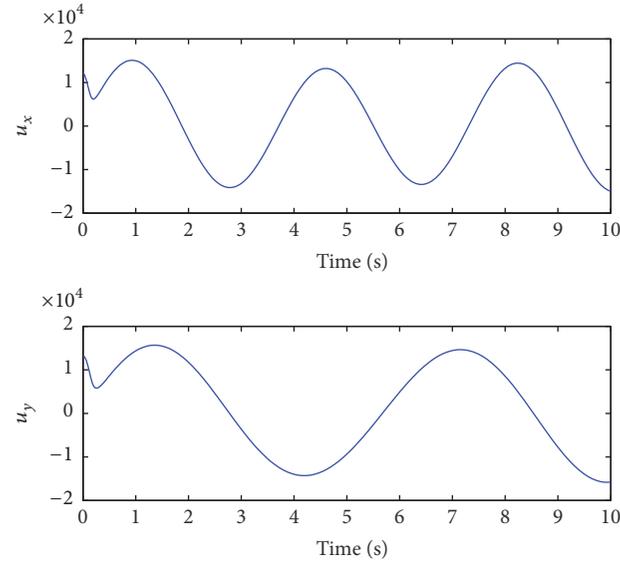


FIGURE 3: Control inputs.

Then set other simulation parameters as

$$\mathbf{A} = \begin{bmatrix} -0.075 & 0.0025 \\ -0.0175 & -0.0075 \end{bmatrix},$$

$$\mathbf{B} = \begin{bmatrix} -14207 & -877 \\ -877 & -12564 \end{bmatrix},$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

$$\mathbf{K} = \begin{bmatrix} 8 & 0 \\ 0 & 4 \end{bmatrix},$$

$$\boldsymbol{\beta} = \begin{bmatrix} 20 & 0 \\ 0 & 15 \end{bmatrix},$$

$$\gamma = 80,$$

$$\kappa = 0.01,$$

$$\tau = 80,$$

$$\alpha = 0.01.$$

(26)

And select the initial state values of the system as $[0.8 \ 0 \ 1 \ 0]^T$. The centers of basis function for network are uniformly valued in $[-1 \ 1]$, and the spreads of the basis function are $\sigma_i = 1$. The number of neural network nodes is chosen as 256. The adaptive signals are presented in Figure 2, and the control inputs are shown in Figure 3. As

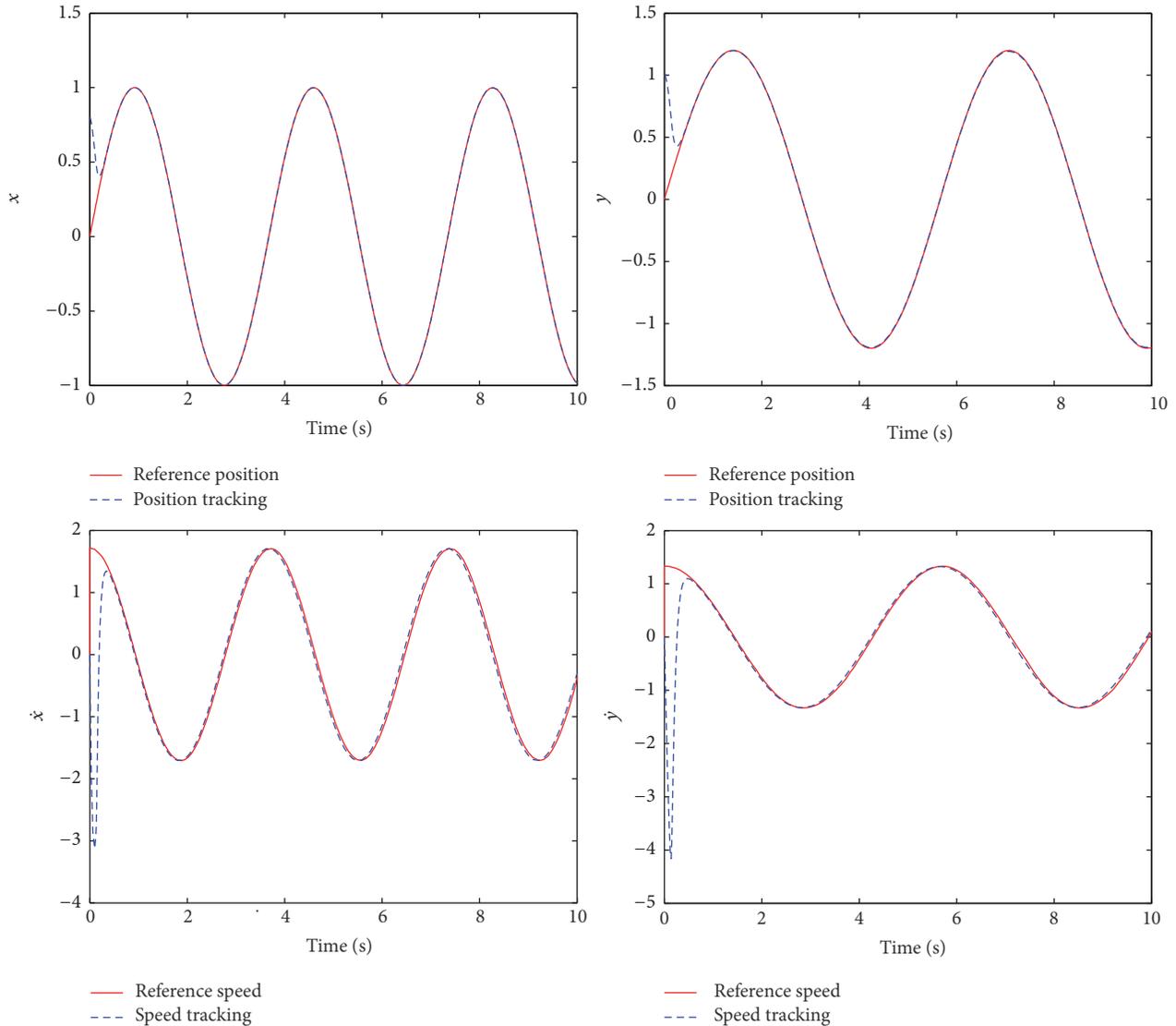


FIGURE 4: Position and speed trajectories.

depicted in Figure 4, the system using adaptive neural network sliding mode control with minimal-learning-parameter technique could track the reference signals very well. The position tracking errors and speed tracking errors are shown in Figure 5. Through the tracking simulations of MEMS gyroscope, the proposed approach has satisfying performance.

6. Conclusions

In this paper, an adaptive neural network sliding mode control strategy is proposed to compensate parametric uncertainties and external disturbances of MEMS gyroscopes. Based on Lyapunov criterion, system stability is guaranteed. With minimal-learning-parameter technique, the online

computation burden is significantly reduced. Numerical simulations verify that the novel control scheme could track reference trajectories very well, which is similar to conventional adaptive neural network sliding mode control scheme. Therefore, the control scheme proposed in this paper could force the mass moves along reference trajectories, so that the performances of MEMS gyroscopes are improved.

For future work, more efficient learning methods [21–25] will be tested on the dynamics while the implementation for real systems will be analyzed.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

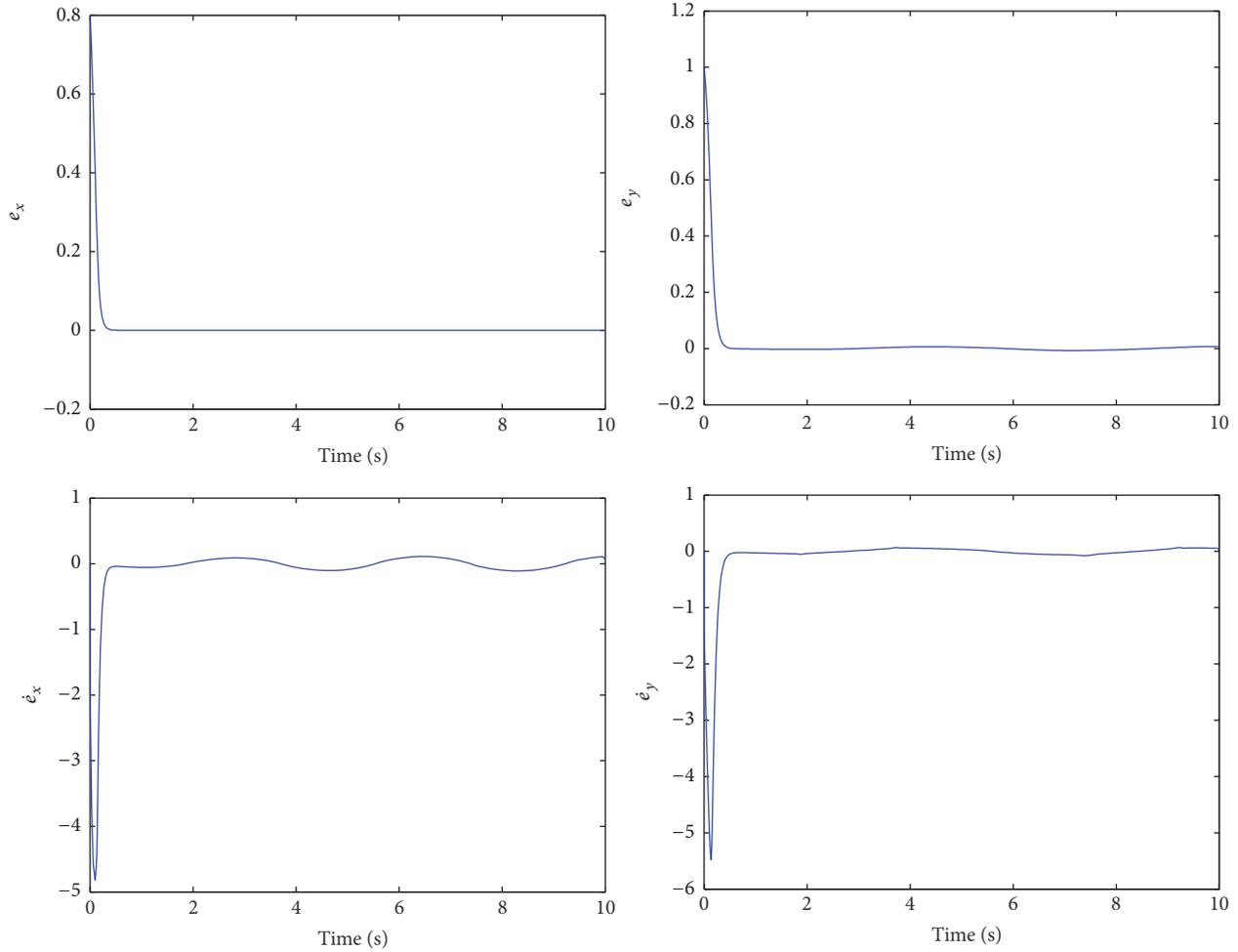


FIGURE 5: Position and speed tracking errors.

Acknowledgments

This work was supported by Industrial Science and Technology Key Project of Shaanxi Province (Grant no. 2016GY-070) and Key Project of Shaanxi Province's Department of Education (Grant no. 2016JS017).

References

- [1] M. Chen and S. S. Ge, "Adaptive neural output feedback control of uncertain nonlinear systems with unknown hysteresis using disturbance observer," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 12, pp. 7706–7716, December 2015.
- [2] B. Xu, D. Wang, Y. Zhang, and Z. Shi, "DOB based neural control of flexible hypersonic flight vehicle considering wind effects," *IEEE Transactions on Industrial Electronics*, vol. PP, no. 99, p. 1, 2017.
- [3] B. Xu, "Composite learning finite-time control with application to quadrotors," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–10, 2017.
- [4] G.-X. Wen, C. L. Chen, Y.-J. Liu, and Z. Liu, "Neural-network-based adaptive leader-following consensus for second-order non-linear multi-agent systems," *IET Control Theory and Applications*, vol. 9, no. 13, pp. 1927–1934, 2015.
- [5] C. Yang, X. Wang, Z. Li, Y. Li, and C. Su, "Teleoperation control based on combination of wave variable and neural networks," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–12, 2016.
- [6] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 3, pp. 1162–1171, 2016.
- [7] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–12, 2016.
- [8] Q. Yang, S. Jagannathan, and Y. Sun, "Robust integral of neural network and error sign control of MIMO nonlinear systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3278–3286, 2015.
- [9] S. Park and R. Horowitz, "New adaptive mode of operation for MEMS gyroscopes," *Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME*, vol. 126, no. 4, pp. 800–810, 2004.
- [10] R. P. Leland, "Adaptive control of a MEMS gyroscope using lyapunov methods," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 2, pp. 278–283, 2006.
- [11] J. D. John and T. Vinay, "Novel concept of a single-mass adaptively controlled triaxial angular rate sensor," *IEEE Sensors Journal*, vol. 6, no. 3, pp. 588–595, 2006.

- [12] C. Batur, T. Sreeramreddy, and Q. Khasawneh, "Sliding mode control of a simulated MEMS gyroscope," *ISA Transactions*, vol. 45, no. 1, pp. 99–108, 2006.
- [13] A. Ebrahimi, "Regulated model-based and non-model-based sliding mode control of a MEMS vibratory gyroscope," *Journal of Mechanical Science and Technology*, vol. 28, no. 6, pp. 2343–2349, 2014.
- [14] J. Fei and C. Batur, "A novel adaptive sliding mode control with application to MEMS gyroscope," *ISA Transactions*, vol. 48, no. 1, pp. 73–78, 2009.
- [15] F.-J. Lin, S.-Y. Chen, and K.-K. Shyu, "Robust dynamic sliding-mode control using adaptive RENN for magnetic levitation system," *IEEE Transactions on Neural Networks*, vol. 20, no. 6, pp. 938–951, 2009.
- [16] L. Y. Wang, T. Y. Chai, and L. F. Zhai, "Neural-network-based terminal sliding-mode control of robotic manipulators including actuator dynamics," *IEEE Transactions on Industrial Electronics*, vol. 56, no. 9, pp. 3296–3304, 2009.
- [17] J. Fei and Y. Yang, "Adaptive neural compensation scheme for robust tracking of MEMS gyroscope," in *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2012*, pp. 1546–1551, October 2012.
- [18] Y. P. Asad, A. Shamsi, and J. Tavoosi, "Backstepping-based recurrent type-2 fuzzy sliding mode control for MIMO systems (MEMS triaxial gyroscope case study)," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 25, no. 2, pp. 213–233, 2017.
- [19] J. X. Ren, R. Zhang, and B. Xu, "Adaptive fuzzy sliding mode control of MEMS gyroscope with finite time convergence," *Journal of Sensors*, vol. 2016, Article ID 1572303, 7 pages, 2016.
- [20] B. Xu, Y. Fan, and S. Zhang, "Minimal-learning-parameter technique based adaptive neural control of hypersonic flight dynamics without back-stepping," *Neurocomputing*, vol. 164, no. 1-2, pp. 201–209, 2015.
- [21] J. Yang, S. Li, and X. Yu, "Sliding-mode control for systems with mismatched uncertainties via a disturbance observer," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 1, pp. 160–169, 2013.
- [22] B. Xu and F. Sun, "Composite intelligent learning control of strict-feedback systems with disturbance," *IEEE Transactions on Cybernetics*, vol. PP, no. 99, pp. 1–12, 2017.
- [23] M. Wang, Y. Zhang, and H. Ye, "Dynamic learning from adaptive neural control of uncertain robots with guaranteed full-state tracking precision," *Complexity*, In press.
- [24] B. Luo, T. Huang, H.-N. Wu, and X. Yang, "Data-driven H_∞ control for nonlinear distributed parameter systems," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 11, pp. 2949–2961, 2015.
- [25] B. Xu and P. Zhang, "Composite learning sliding mode control of flexible-link manipulator," *Complexity*, In press.

Research Article

Pinning Synchronization for Complex Networks with Interval Coupling Delay by Variable Subintervals Method and Finsler's Lemma

Dawei Gong,¹ Frank L. Lewis,^{2,3} Liping Wang,⁴ Dong Dai,⁵ and Shuang Zhang⁶

¹School of Mechatronics Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

²UTA Research Institute, The University of Texas at Arlington, Arlington, TX 76118, USA

³Northeastern University, Shenyang 110036, China

⁴Department of Mechanical Engineering, Tsinghua University, Beijing 100084, China

⁵School of Electric Power, South China University of Technology, Guangzhou 510641, China

⁶School of Astronautics and Aeronautics, University of Electronic Science and Technology of China, Chengdu 611731, China

Correspondence should be addressed to Dawei Gong; pzhzhx@126.com

Received 18 March 2017; Accepted 4 May 2017; Published 8 June 2017

Academic Editor: Junpei Zhong

Copyright © 2017 Dawei Gong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The pinning synchronous problem for complex networks with interval delays is studied in this paper. First, by using an inequality which is introduced from Newton-Leibniz formula, a new synchronization criterion is derived. Second, combining Finsler's Lemma with homogenous matrix, convergent linear matrix inequality (LMI) relaxations for synchronization analysis are proposed with matrix-valued coefficients. Third, a new variable subintervals method is applied to expand the obtained results. Different from previous results, the interval delays are divided into some subdelays, which can introduce more free weighting matrices. Fourth, the results are shown as LMI, which can be easily analyzed or tested. Finally, the stability of the networks is proved via Lyapunov's stability theorem, and the simulation of the trajectory claims the practicality of the proposed pinning control.

1. Introduction

Complex networks have large size and nontrivial complex topological features have been intensively studied by many researchers in recent years. Such networks have connections which are neither purely regular nor purely random. These networks are used to understand and predict the behavior of many structures, for example, Internet, medicine, society, and biology.

It has been found that lots of phenomena in real world can be studied by complex networks (such as [1–5] and references therein). Amongst all the topics which are studied by complex networks, synchronization phenomena play an important role due to their real world potential applications. There are many interesting synchronization phenomena in nature world. Lots of efforts have been put into the development of the synchronization problems in complex networks [6–11].

It should be noticed that time-varying delays occur commonly in connection topology of networks which are more realistic and cover more situations in practice. Therefore, various kinds of delay methods have been proposed, and synchronization problems for networks with delay have been extensively studied [12–14]. However, the methods to deal with the delay in these papers always need large amount of calculation. So how to remove the redundant computation and improve networks' performance is still a challenging objective.

Normally, complex networks cannot synchronize by themselves, and some controllers are designed to force the system to be synchronized. However, it is hard to design or realize controllers for all nodes of large network structure. Therefore, pinning controllers have been widely used to synchronize complex networks. In [15], an adaptive predictive pinning control is proposed to suppress the cascade in

coupled map lattices (CMLs). In [16], by using piecewise Lyapunov theory, some less conservative criteria are deduced for exponential synchronization of the complex networks. In [17], a new adaptive intermittent scheme is used to deduce some novel criteria by utilizing a piecewise auxiliary and other relative references [18–21]. However, in the above papers, many useful situations such as some novel delay processing methods and Finsler's Lemma which can introduce more matrix-valued coefficients to synchronization criteria are not utilized. As far as I know, such pinning synchronization methodology for complex networks has not been proposed yet.

Motivated by the former discussions, we elaborate pinning synchronization results for complex networks via subintervals delay method and Finsler's Lemma. By constructing a novel Lyapunov-Krasovskii function (LKF) and using some mathematical skills proposed in this paper, complex networks can achieve synchronization.

Notations

- $R^n \leftrightarrow n$ -dimensional Euclidean space
- $R^{m \times n} \leftrightarrow m \times n$ real matrices
- $I_n \leftrightarrow n$ -dimensional identity matrix
- $A \otimes B \leftrightarrow$ Kronecker product of matrices A and B
- $\text{diag}(\dots) \leftrightarrow$ block-diagonal matrix
- $\begin{bmatrix} X & Y \\ * & Z \end{bmatrix} \leftrightarrow \begin{bmatrix} X & Y \\ Y^T & Z \end{bmatrix}$

2. Preliminaries

Consider the system which consists of N nodes, and each node has an n -dimensional subsystem; then the pinning control system can be written as

$$\dot{x}_i(t) = f(x_i(t)) + c \sum_{j=1}^N g_{ij} A x_j(t - \tau(t)) + u_i, \quad (1)$$

$$i = 1, 2, \dots, l,$$

$$\dot{x}_i(t) = f(x_i(t)) + c \sum_{j=1}^N g_{ij} A x_j(t - \tau(t)), \quad (2)$$

$$i = l + 1, l + 2, \dots, N,$$

$x_i(t) = (x_{i1}(t), x_{i2}(t), \dots, x_{in}(t))^T \in R^n$. $f(x_i(t)) \in R^n$ is the activation function. The constants c ($c > 0$), $A = (a_{ij})_{n \times n}$, and $G = (g_{ij})_{N \times N}$ are, respectively, representing coupling strength, inner-coupling matrix, and outer-coupling connections.

$\tau(t)$ satisfies

$$0 \leq \tau(t) \leq \tau, \quad (3)$$

$$\dot{\tau}(t) \leq \mu < 1.$$

u_i , $i = 1, 2, \dots, N$, are the pinning controllers, which are designed as

$$u_i(t) = -c\eta_i A (x_i(t - \tau(t)) - s(t - \tau(t))) \in R^n. \quad (4)$$

Let $\eta_i > 0$, for $i = 1, 2, \dots, l$, and $\eta_i = 0$, for $i = l + 1, l + 2, \dots, N$, η_i are the control gains. Then we can get

$$\begin{aligned} \dot{x}_i(t) &= f(x_i(t)) - f(s(t)) \\ &+ c \sum_{j=1}^N g_{ij} A (x_j(t - \tau(t)) - s(t - \tau(t))) \\ &- c\eta_i A (x_i(t - \tau_i) - s(t - \tau_i)), \end{aligned} \quad (5)$$

$$i = 1, 2, \dots, N.$$

In this following, we will introduce some elementary situations.

Assumption 1. The outer-coupling matrix G satisfies

$$g_{ij} = g_{ji} \geq 0, \quad i \neq j$$

$$g_{ii} = - \sum_{j=1, j \neq i}^N g_{ij}, \quad i, j = 1, 2, \dots, N. \quad (6)$$

Definition 2. System (2) is synchronized if

$$x_1(t) = \dots = x_N(t) = s(t), \quad t \rightarrow \infty, \quad (7)$$

where $\dot{s}(t) = f(s(t))$ and $s(t)$ is a solution of an isolate node.

Lemma 3 (see [22]). *The eigenvalues of G in system (2) is defined by*

$$\lambda_N \leq \lambda_{N-1} \leq \dots \leq \lambda_3 \leq \lambda_2 \leq \lambda_1 = 0. \quad (8)$$

On the other hand, if $N - 1$ of n -dimensional differential equations of their 0 solution are asymptotically stable

$$\dot{w}_k(t) = c\lambda_k A w_k(t - \tau(t)) + J(t) w_k(t). \quad (9)$$

The Jacobian matrix of $f(x(t))$ at $s(t)$ is $J(t)$; then the synchronized states (2) are the same as the asymptotically stable results of system (9).

Lemma 4 (Jensen's inequality). *For positive definite symmetric matrices $\omega \in R^{n \times n}$, $\omega : [0, \rho] \rightarrow R^n$, scalar $\rho > 0$, we have*

$$\begin{aligned} &\rho \int_0^\rho \bar{\omega}^T(s) \Upsilon \omega(s) ds \\ &\geq \left(\int_0^\rho \omega(s) ds \right)^T \Upsilon \left(\int_0^\rho \omega(s) ds \right). \end{aligned} \quad (10)$$

Lemma 5 (Finsler's Lemma). *Let $\zeta \in R^n$, $\phi^T = \phi \in R^{n \times n}$, and $\mathfrak{B} \in R^{m \times n}$, $\text{rank}(\mathfrak{B}) < n$. Then*

- (1) $\zeta^T \phi \zeta < 0$, $\mathfrak{B} \zeta = 0$, $\zeta \neq 0$
- (2) $(\mathfrak{B}^\perp)^T \phi \mathfrak{B}^\perp < 0$,
- (3) $\exists L \in R^{n \times m}$, $\Phi + L \mathfrak{B} + \mathfrak{B}^T L^T < 0$,

where \mathfrak{B}^\perp is a right orthogonal complement of \mathfrak{B} .

Lemma 6 (see [23]). *Let $x(t) \in \mathbb{R}^n$, $M_1, M_2 \in \mathbb{R}^{n \times n}$, $\chi^T = \chi > 0 \in \mathbb{R}^{n \times n}$, $Z \in \mathbb{R}^{2n \times 2n}$, a constant $\tau \geq 0$, and then the integral inequality is defined as follows:*

$$-\int_{t-\tau}^t \dot{x}^T(s) \chi \dot{x}(s) ds \leq \xi^T(t) \Upsilon \xi(t) + \tau \xi^T(t) Z \xi(t), \quad (11)$$

where

$$\begin{aligned} \Upsilon &:= \begin{bmatrix} M_1^T + M_1 & -M_1^T + M_2 \\ * & -M_2^T - M_2 \end{bmatrix}, \\ \xi(t) &:= \begin{bmatrix} x(t) \\ x(t-\tau) \end{bmatrix}, \\ Z &= \begin{bmatrix} M_1^T \\ M_2^T \end{bmatrix} \chi^{-1} [M_1 \ M_2]. \end{aligned} \quad (12)$$

3. Main Results

Theorem 7. *For positive definite symmetric matrices P_k , Q_k , and S_k , real matrices M_{ik} , ($i = 1, 2$), and the following LMIs hold for all $2 \leq k \leq N$:*

$$\Xi_{1k} = \begin{bmatrix} H_{1k} & \tau \Gamma_1^T S_k & \tau(1-\mu) \Gamma_2^T \\ * & -\tau S_k & 0 \\ * & * & -\tau(1-\mu) S_k \end{bmatrix} < 0, \quad (13)$$

where

$$\begin{aligned} \Gamma_1 &= [J(t), c\lambda_k A], \\ \Gamma_2 &= [M_{1k}, M_{2k}], \\ H_{1k} &= \begin{bmatrix} \Omega_{11} & \Omega_{12} \\ * & \Omega_{22} \end{bmatrix}, \\ \Omega_{11} &= P_k J(t) + J(t)^T P_k + Q_k + (1-\mu)(M_{1k}^T + M_{1k}), \\ \Omega_{12} &= P_k c\lambda_k A - (1-\mu)(M_{1k}^T - M_{2k}), \\ \Omega_{22} &= -(1-\mu)Q_k - (1-\mu)(M_{2k}^T + M_{2k}). \end{aligned} \quad (14)$$

Then system (2) is synchronized.

Proof. The Lyapunov function is confined in the following:

$$\begin{aligned} V(t) &= w_k^T(t) P_k w_k(t) + \int_{t-\tau(t)}^t w_k^T(s) Q_k w_k(s) ds \\ &+ \int_{t-\tau(t)}^t \int_{\theta}^t \dot{w}_k^T(s) S_k \dot{w}_k(s) ds d\theta. \end{aligned} \quad (15)$$

Then $\dot{V}(t)$ can be expressed as

$$\begin{aligned} \dot{V}(t) &= 2w_k^T(t) P_k [J(t) w_k(t) + c\lambda_k A w_k(t-\tau(t))] \\ &+ w_k^T(t) Q_k w_k(t) \\ &- (1-\dot{\tau}(t)) w_k^T(t-\tau(t)) Q_k w_k(t-\tau(t)) \\ &+ \tau(t) \dot{w}_k^T(t) S_k \dot{w}_k(t) \\ &- (1-\dot{\tau}(t)) \int_{t-\tau(t)}^t \dot{w}_k^T(s) S_k \dot{w}_k(s) ds \\ &\leq 2w_k^T(t) P_k [J(t) w_k(t) + c\lambda_k A w_k(t-\tau(t))] \\ &+ w_k^T(t) Q_k w_k(t) \\ &- (1-\mu) w_k^T(t-\tau(t)) Q_k w_k(t-\tau(t)) \\ &+ \tau \dot{w}_k^T(t) S_k \dot{w}_k(t) \\ &- (1-\mu) \int_{t-\tau(t)}^t \dot{w}_k^T(s) S_k \dot{w}_k(s) ds. \end{aligned} \quad (16)$$

From Lemma 6, for any constant matrices M_{1k}, M_{2k}

$$\begin{aligned} &-\int_{t-\tau(t)}^t \dot{w}_k^T(s) S_k \dot{w}_k(s) ds \\ &\leq \eta_k^T(t) \begin{bmatrix} M_{1k}^T + M_{1k} & -M_{1k}^T + M_{2k} \\ * & -M_{2k}^T - M_{2k} \end{bmatrix} \eta_k(t) \\ &+ \tau \eta_k^T(t) \begin{bmatrix} M_{1k}^T \\ M_{2k}^T \end{bmatrix} S_k^{-1} [M_{1k} \ M_{2k}] \eta_k(t), \end{aligned} \quad (17)$$

where

$$\eta_k^T(t) = [w_k^T(t), w_k^T(t-\tau(t))]. \quad (18)$$

Then

$$\begin{aligned} \dot{V}(t) &\leq \eta_k^T(t) (H_{1k} + \tau \Gamma_1^T S_k \Gamma_1 + \tau(1-\mu) \Gamma_2^T S_k^{-1} \Gamma_2) \eta_k(t). \end{aligned} \quad (19)$$

By Schur complement, $H_{1k} + \tau \Gamma_1^T S_k \Gamma_1 + \tau(1-\mu) \Gamma_2^T S_k^{-1} \Gamma_2$ is equivalent to expression (13). Then the proof is completed. \square

In the following criteria, we will introduce Finsler's Lemma. Combining with the Finsler's Lemma, convergent LMI relaxations for synchronization analysis are proposed.

Theorem 8. *From Lemma 5, dynamical system (2) is asymptotically synchronized if there exist $P_k = P_k^T > 0$, $Q_k = Q_k^T > 0$, $S_k = S_k^T > 0$, $Y_k = Y_k^T > 0$, and any real matrices M_{ik} , ($i = 1, 2$), and the following LMIs hold for all $2 \leq k \leq N$:*

$$(\mathbb{B}^\pm)^T \Xi_{2k} \mathbb{B}^\pm < 0, \quad (20)$$

where

$$\begin{aligned} \mathbb{B} &= [J(t), c\lambda_k A, -I_N], \\ \Xi_{2k} &= \begin{bmatrix} H_{2k} & \tau\Gamma_1^T Y_k^T & \tau(1-\mu)\Gamma_2^T \\ * & -Y_k - Y_k^T + \tau S_k & 0 \\ * & * & -\tau(1-\mu)S_k \end{bmatrix} < 0, \\ \Gamma_1 &= [J(t), c\lambda_k A], \\ \Gamma_2 &= [M_{1k}, M_{2k}], \\ H_{2k} &= \begin{bmatrix} \Omega_{11} & \Omega_{12} \\ * & \Omega_{22} \end{bmatrix}. \\ \Omega_{11} &= P_k J(t) + J(t)^T P_k + Q_k + (1-\mu)(M_{1k}^T + M_{1k}) \\ \Omega_{12} &= P_k c\lambda_k A - (1-\mu)(M_{1k}^T - M_{2k}) \\ \Omega_{22} &= -(1-\mu)Q_k - (1-\mu)(M_{2k}^T + M_{2k}). \end{aligned} \quad (21)$$

Proof. Choose the same LKF in Theorem 7. From system (9), the following equation holds for any matrices Y_k ($2 \leq k \leq N$),

$$0 = 2\dot{w}_k^T(t) \cdot Y_k [-\dot{w}_k(t) + J(t)w_k(t) + c\lambda_k A w_k(t - \tau(t))]. \quad (22)$$

Combing (16), (17), and (22), we can obtain

$$\dot{V}_k(t) \leq \zeta_k^T(t) \Xi_{2k} \zeta_k(t), \quad (23)$$

where

$$\zeta_k^T(t) = [w_k^T(t), w_k^T(t - \tau(t)), \dot{w}_k^T(t)]. \quad (24)$$

Note $\mathbb{B}\zeta_k(t) = 0$; it follows from Lemma 5 that $(\mathbb{B}^\perp)^T \Xi_{2k} \mathbb{B}^\perp < 0$ is equivalent to $\zeta_k^T(t) \Xi_{2k} \zeta_k(t) < 0$. \square

Remark 9. Convergent LMI relaxations are introduced by Finsler's Lemma with homogenous matrix. Then more matrix-valued coefficients can be introduced to reduce conservatism. Moreover, our methods can also be applied to most of the existing synchronization results, such as [6–21].

Remark 10. Different from [24, 25] which divide the constant delay part into many more same size delay, the interval $[0, \tau(t)]$ can be chosen arbitrarily into smaller variable subintervals $[0, \rho\tau(t)]$, $[\rho\tau(t), \tau(t)]$, where $\rho \in (0, 1)$.

Theorem 11. From Lemma 5, for given scalar $0 < \rho < 1$, system (2) is synchronized if there exist $P_k = P_k^T > 0$, $Q_k = Q_k^T > 0$, $R_k = R_k^T > 0$, $W_k = W_k^T > 0$, $S_{ik} = S_{ik}^T > 0$, ($i = 1, 2$), and real matrices Y_k , M_{ik} , and T_{ik} , ($i = 1, 2$), and the following LMIs hold for all $2 \leq k \leq N$:

$$(\mathbb{B}^\perp)^T \Xi_{3k} \mathbb{B}^\perp < 0, \quad (25)$$

where

$$\begin{aligned} \mathbb{B} &= [J(t), c\lambda_k A, 0, 0, -I_N], \\ \Xi_{3k} &= \begin{bmatrix} H_{3k}^{(1)} & 0 & \tau\Gamma_1^T Y_k^T & \tau(1-\rho\mu)\Gamma_2^T & 0 \\ * & H_{3k}^{(2)} & 0 & 0 & \tau(1-\rho)(1-\rho\mu)\Gamma_3^T \\ * & * & H_{3k}^{(3)} & 0 & 0 \\ * & * & * & -\tau(1-\rho\mu)S_{1k} & 0 \\ * & * & * & * & -\tau(1-\rho)(1-\rho\mu)S_{2k} \end{bmatrix} < 0, \\ H_{3k}^{(1)} &= \begin{bmatrix} \Omega_{11}^{(1)} & \Omega_{12}^{(1)} \\ * & \Omega_{22}^{(1)} \end{bmatrix}. \\ \Omega_{11}^{(1)} &= P_k J(t) + J(t)^T P_k + Q_k + R_k + W_k + (1-\rho\mu)(M_{1k}^T + M_{1k}) \\ \Omega_{12}^{(1)} &= P_k c\lambda_k A - (1-\rho\mu)(M_{1k}^T - M_{2k}) \\ \Omega_{22}^{(1)} &= -(1-\mu)Q_k - (1-\mu)(M_{2k}^T + M_{2k}) \\ H_{3k}^{(2)} &= \begin{bmatrix} \Omega_{11}^{(2)} & \Omega_{12}^{(2)} \\ * & \Omega_{22}^{(2)} \end{bmatrix}. \\ \Omega_{11}^{(2)} &= (1-\rho\mu)(T_{1k}^T + T_{1k}) - (1-\rho\mu)R_k \\ \Omega_{12}^{(2)} &= -(1-\rho\mu)(T_{1k}^T - T_{2k}) \end{aligned}$$

$$\begin{aligned}
\Omega_{22}^{(2)} &= -(1 - \rho\mu)(T_{2k}^T + T_{2k}) - W_k. \\
\Gamma_1 &= [J(t), c\lambda_k A], \\
\Gamma_2 &= [M_{1k}, M_{2k}], \\
\Gamma_3 &= [T_{1k}, T_{2k}] \\
H_{3k}^{(3)} &= -Y_k - Y_k^T + \rho\tau S_{1k} + (\tau - \rho\tau) S_{2k}.
\end{aligned} \tag{26}$$

Proof. The LKF is confined in the following inequality:

$$V_k(t) = V_{1k}(t) + V_{2k}(t) + V_{3k}(t) + V_{4k}(t) + V_{5k}(t), \quad (27)$$

where

$$\begin{aligned}
V_{1k}(t) &= w_k^T(t) P_k w_k(t) \\
V_{2k}(t) &= \int_{t-\tau(t)}^t w_k^T(s) Q_k w_k(s) ds \\
V_{3k}(t) &= \int_{t-\rho\tau(t)}^t w_k^T(s) R_k w_k(s) ds \\
&\quad + \int_{t-\tau}^t w_k^T(s) W_k w_k(s) ds \\
V_{4k}(t) &= \int_{t-\rho\tau(t)}^t \int_{\theta}^t \dot{w}_k^T(s) S_{1k} \dot{w}_k(s) ds d\theta \\
V_{5k}(t) &= \int_{t-\tau}^{t-\rho\tau(t)} \int_{\theta}^t \dot{w}_k^T(s) S_{2k} \dot{w}_k(s) ds d\theta.
\end{aligned} \tag{28}$$

Then $\dot{V}(t)$ can be expressed as

$$\begin{aligned}
\dot{V}_{1k}(t) &= 2w_k^T(t) P_k [J(t) w_k(t) + c\lambda_k A w_k(t - \tau(t))] \\
\dot{V}_{2k}(t) &= w_k^T(t) Q_k w_k(t) \\
&\quad - (1 - \dot{\tau}(t)) w_k^T(t - \tau(t)) Q_k w_k(t - \tau(t)) \\
\dot{V}_{3k}(t) &= w_k^T(t) R_k w_k(t) \\
&\quad - (1 - \rho\dot{\tau}(t)) w_k^T(t - \rho\tau(t)) R_k w_k(t - \rho\tau(t)) \\
&\quad + w_k^T(t) W_k w_k(t) - w_k^T(t - \tau) W_k w_k(t - \tau) \\
\dot{V}_{4k}(t) &= \rho\tau \dot{w}_k^T(t) S_{1k} \dot{w}_k(t) \\
&\quad - (1 - \rho\dot{\tau}(t)) \int_{t-\tau(t)}^t \dot{w}_k^T(s) S_{1k} \dot{w}_k(s) ds
\end{aligned}$$

$$\begin{aligned}
\dot{V}_{5k}(t) &= (\tau - \rho\tau(t)) \dot{w}_k^T(t) S_{2k} \dot{w}_k(t) \\
&\quad - (1 - \rho\dot{\tau}(t)) \int_{t-\tau(t)}^{t-\rho\tau(t)} \dot{w}_k^T(s) S_{2k} \dot{w}_k(s) ds.
\end{aligned} \tag{29}$$

From Lemma 6, for any constant matrices M_{1k}, M_{2k}

$$\begin{aligned}
& - \int_{t-\tau(t)}^t \dot{w}_k^T(s) S_{1k} \dot{w}_k(s) ds \\
& \leq \eta_{1k}^T(t) \begin{bmatrix} M_{1k}^T + M_{1k} & -M_{1k}^T + M_{2k} \\ * & -M_{2k}^T - M_{2k} \end{bmatrix} \eta_{1k}(t) \\
& \quad + \tau \eta_{1k}^T(t) \begin{bmatrix} M_{1k}^T \\ M_{2k}^T \end{bmatrix} S_{1k}^{-1} [M_{1k} \ M_{2k}] \eta_{1k}(t),
\end{aligned} \tag{30}$$

where

$$\eta_{1k}^T(t) = [w_k^T(t), w_k^T(t - \tau(t))]. \tag{31}$$

For any appropriate dimension matrices T_{1k}, T_{2k} , we have

$$\begin{aligned}
& - \int_{t-\tau}^{t-\rho\tau(t)} \dot{w}_k^T(s) S_{2k} \dot{w}_k(s) ds \\
& \leq \eta_{2k}^T(t) \begin{bmatrix} T_{1k}^T + T_{1k} & -T_{1k}^T + T_{2k} \\ * & -T_{2k}^T - T_{2k} \end{bmatrix} \eta_{2k}(t) \\
& \quad + \tau(1 - \rho) \eta_{2k}^T(t) \begin{bmatrix} T_{1k}^T \\ T_{2k}^T \end{bmatrix} S_{2k}^{-1} [T_{1k} \ T_{2k}] \eta_{2k}(t),
\end{aligned} \tag{32}$$

where

$$\eta_{2k}^T(t) = [w_k^T(t - \rho\tau(t)), w_k^T(t - \tau)]. \tag{33}$$

From system (9), the following equation holds for any matrices Y_k ($2 \leq k \leq N$):

$$\begin{aligned}
0 &= 2\dot{w}_k^T(t) \\
&\quad \cdot Y_k [-\dot{w}_k(t) + J(t) w_k(t) + c\lambda_k A w_k(t - \tau(t))].
\end{aligned} \tag{34}$$

Substituting the proposed equalities,

$$\dot{V}_k(t) \leq \Theta_k^T(t) \Xi_{3k} \Theta_k(t), \tag{35}$$

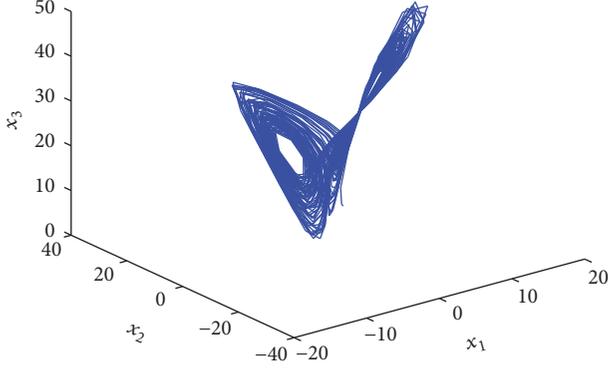


FIGURE 1: The chaotic trajectory of Lorenz systems.

where

$$\Theta_k^T(t) = [w_k^T(t), w_k^T(t - \tau(t)), w_k^T(t - \rho\tau(t)), w_k^T(t - \tau), \dot{w}_k^T(t)]. \quad (36)$$

From Lemma 5, $\Theta_k^T(t)\Xi_{3k}\Theta_k(t) < 0$ can be acquired. This completes the proof. \square

Remark 12. Obviously, when we choose different values of ρ , the synchronization criteria can also be changed. Through the choice of appropriate parameters ρ , different stability results can be obtained.

Remark 13. By using Lemma 6 and introducing Finsler's Lemma, some delay-dependent conditions are acquired in complex networks. The criteria in this paper can be easily used in many existing references and obtain better results, such as [26–30].

4. Numerical Example

Consider the Lorenz system in this example

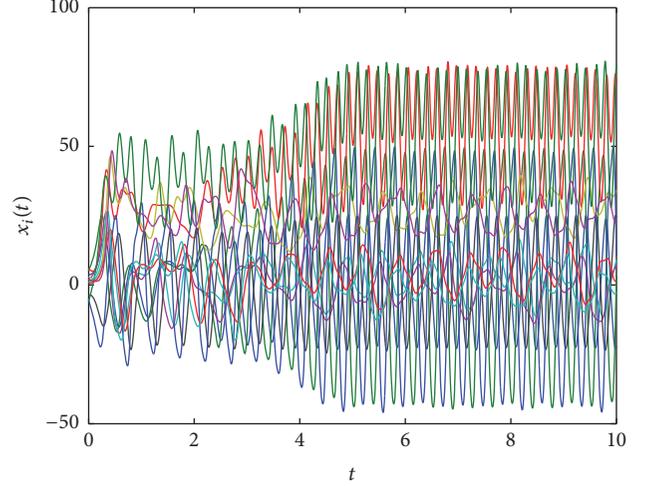
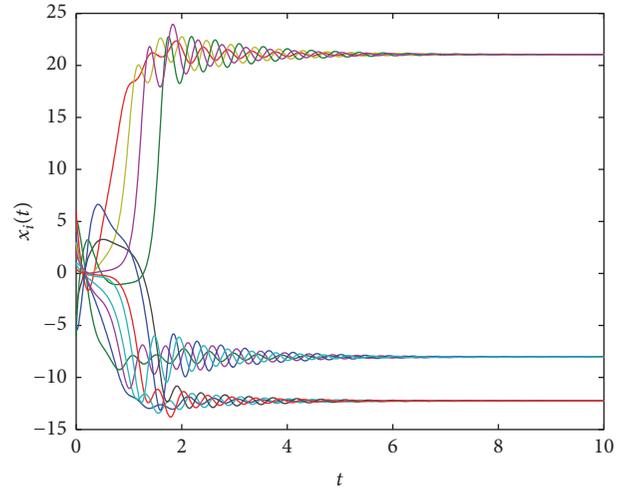
$$\begin{aligned} \dot{x}_1 &= a(x_2 - x_1) \\ \dot{x}_2 &= \hat{c}x_1 - x_1x_3 - x_2 \\ \dot{x}_3 &= x_1x_2 - bx_3, \end{aligned} \quad (37)$$

where $a > 0$, $b > 0$, $\hat{c} > 0$. When $a = 10$, $b = 8/3$, $\hat{c} = 28$, the net is chaotic, and its behavior is shown in Figure 1.

We assume $c = 1$, and the derivatives of time delays are $\mu = 0$, $\tau = 0.2$, $\Lambda = 6 * \text{diag}(1, 1, 1, 1)$, and its Jacobian is

$$J(t) = \begin{bmatrix} -a & a & 0 \\ \hat{c} & -1 & 0 \\ 0 & 0 & -b \end{bmatrix},$$

$$G = \begin{bmatrix} -3 & 1 & 1 & 1 \\ 1 & -3 & 1 & 1 \\ 1 & 1 & -3 & 1 \\ 1 & 1 & 1 & -3 \end{bmatrix},$$

FIGURE 2: States of network (2) without control: $x_i(t)$, $i = 1, 2, 3, 4$.FIGURE 3: States of network (2) with control: $x_i(t)$, $i = 1, 2, 3, 4$.

$$A = \begin{bmatrix} 2.4 & 0 & 2.4 \\ 0 & 4.8 & 0 \\ 2.4 & 0 & 2.4 \end{bmatrix}. \quad (38)$$

Figures 2 and 3 show the states of network (2) without and with control, respectively. Figure 4 shows the synchronization errors between every node, $e_j(t) = x_{1j}(t) - x_{ij}(t)$, for $i = 2, 3, 4$, $j = 1, 2, 3$. From Figures 3 and 4, it is easy to see that the system cannot be synchronized by itself, and it can achieve synchronization by pinning control.

5. Conclusion

During the past decades, there has been a rapid development of the techniques about complex networks. Numerous studies have shown that complex network is good at dealing with the problem of function approximation and uncertainties. In the paper, a novel analytical method is provided to ensure

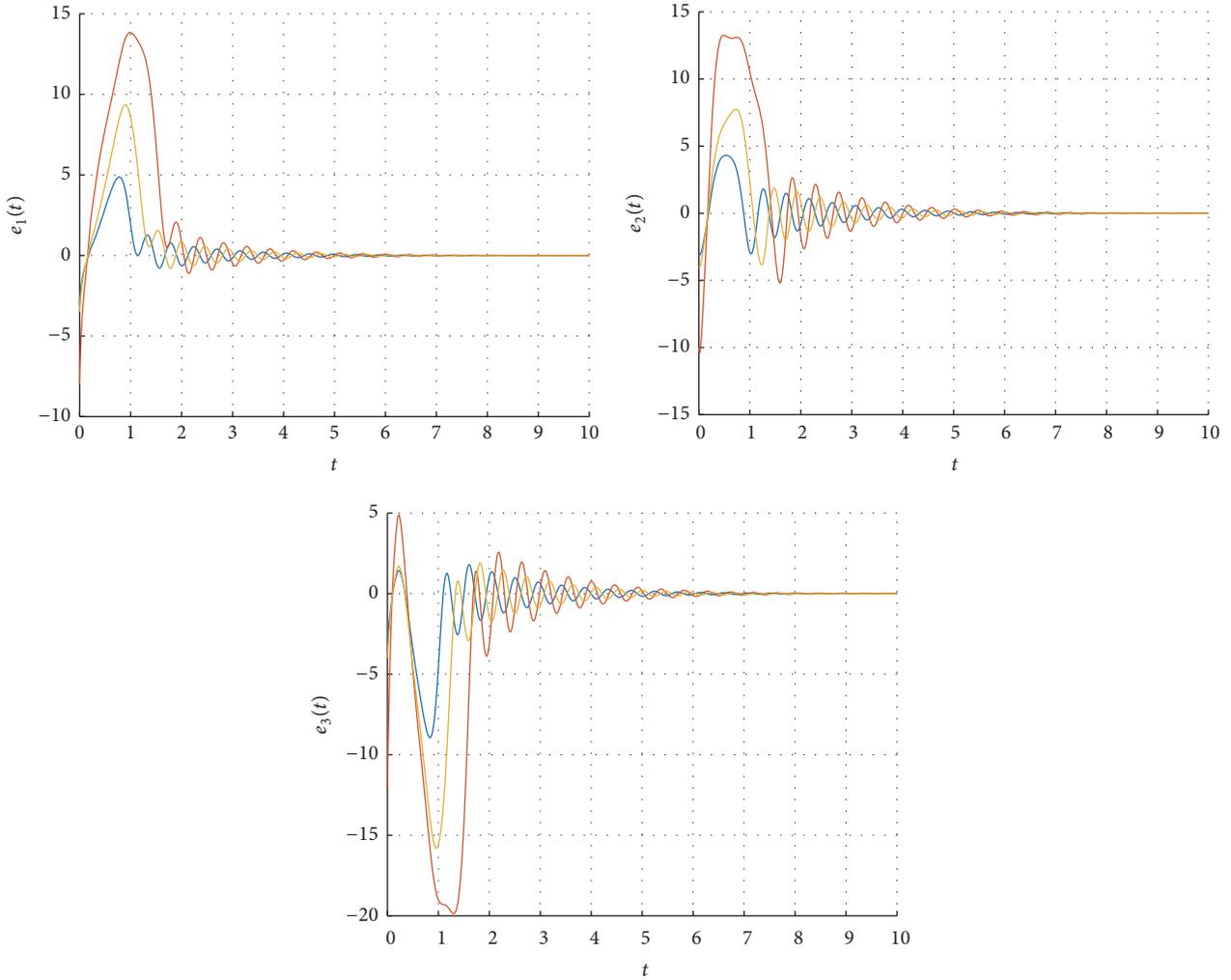


FIGURE 4: Synchronization errors in an example: $e_j(t) = x_{1j}(t) - x_{ij}(t)$ for $i = 2, 3, 4$, $j = 1, 2, 3$.

the synchronization rigorously for complex system. By using Newton-Leibniz formula and introducing Finsler's Lemma, the obtained synchronization criteria are divided in terms of LMI inequalities, and such method has not been obtained until now. On the other hand, in the studies of network-based motion control in actual models [31–34], a novel integral barrier function is first employed for control design of the constrained distributed parameter system modeled as PDEs [35–38], fault tolerance control in complex systems [39], have been hot topics in recent time. Therefore, how to extend our results into these control fields is still a challenging problem.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Science Fund for Distinguished Young Scholar (Grant no. 51225503), the

National Natural Science Foundation of China (51305066 and 61603076), and Fundamental Research Funds for the Central Universities (ZYGX2016J116).

References

- [1] H. Zhang, Z. Wang, and D. Liu, "Global asymptotic stability of recurrent neural networks with multiple time-varying delays," *IEEE Transactions on Neural Networks*, vol. 19, no. 5, pp. 855–873, 2008.
- [2] Z. Wang, H. Zhang, and B. Jiang, "LMI-based approach for global asymptotic stability analysis of recurrent neural networks with various delays and structures," *IEEE Transactions on Neural Networks*, vol. 22, no. 7, pp. 1032–1045, 2011.
- [3] W. He, Y. Chen, and Z. Yin, "Adaptive neural network control of an uncertain robot with full-state constraints," *IEEE Transactions on Cybernetics*, vol. 46, no. 3, pp. 620–629, 2016.
- [4] W. He, Y. Dong, and C. Sun, "Adaptive neural impedance control of a robotic manipulator with input saturation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 3, pp. 334–344, 2016.

- [5] W. He and Y. Dong, "Adaptive fuzzy neural network control for a constrained robot using impedance learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. pp, no. 99, pp. 1–13, 2017.
- [6] H. Gao, J. Lam, and G. Chen, "New criteria for synchronization stability of general complex dynamical networks with coupling delays," *Physics Letters, Section A: General, Atomic and Solid State Physics*, vol. 360, no. 2, pp. 263–273, 2006.
- [7] H. Shen, J. H. Park, Z.-G. Wu, and Z. Zhang, "Finite-time H_∞ synchronization for complex networks with semi-Markov jump topology," *Communications in Nonlinear Science and Numerical Simulation*, vol. 24, no. 1–3, pp. 40–51, 2015.
- [8] H. Kang and X. Fu, "Epidemic spreading and global stability of an SIS model with an infective vector on complex networks," *Communications in Nonlinear Science and Numerical Simulation*, vol. 27, no. 1–3, pp. 30–39, 2015.
- [9] L. Zhang, Y. Wang, and Q. Wang, "Synchronization for time-varying complex dynamical networks with different-dimensional nodes and non-dissipative coupling," *Communications in Nonlinear Science and Numerical Simulation*, vol. 24, no. 1–3, pp. 64–74, 2015.
- [10] D. Tong, W. Zhou, X. Zhou, J. Yang, L. Zhang, and Y. Xu, "Exponential synchronization for stochastic neural networks with multi-delayed and Markovian switching via adaptive feedback control," *Communications in Nonlinear Science and Numerical Simulation*, vol. 29, no. 1-3, pp. 359–371, 2015.
- [11] J. Qin, H. Gao, and W. X. Zheng, "Exponential synchronization of complex networks of linear systems and nonlinear oscillators: a unified analysis," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 3, pp. 510–521, 2015.
- [12] M. Fang, "Synchronization for complex dynamical networks with time delay and discrete-time information," *Applied Mathematics and Computation*, vol. 258, no. 1, pp. 1–11, 2015.
- [13] Y. Liu, Z. Wang, J. Liang, and X. Liu, "Synchronization and state estimation for discrete-time complex networks with distributed delays," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 38, no. 5, pp. 1314–1325, 2008.
- [14] Q. Cheng and J. Cao, "Synchronization of complex dynamical networks with discrete time delays on time scales," *Neurocomputing*, vol. 151, no. 2, pp. 729–736, 2015.
- [15] Z.-J. Bao, G. Wu, and W.-J. Yan, "Control of cascading failures in coupled map lattices based on adaptive predictive pinning control," *Journal of Zhejiang University: Science C*, vol. 12, no. 10, pp. 828–835, 2011.
- [16] H.-Y. Sun, N. Li, D.-P. Zhao, and Q.-L. Zhang, "Synchronization of complex networks with coupling delays via adaptive pinning intermittent control," *International Journal of Automation and Computing*, vol. 10, no. 4, pp. 312–318, 2013.
- [17] C. Hu and H. Jiang, "Pinning synchronization for directed networks with node balance via adaptive intermittent control," *Nonlinear Dynamics*, vol. 80, no. 1-2, pp. 295–307, 2015.
- [18] L. Zhou, C. Wang, and L. Zhou, "Cluster synchronization on multiple sub-networks of complex networks with nonidentical nodes via pinning control," *Nonlinear Dynamics*, vol. 83, no. 1-2, pp. 1079–1100, 2016.
- [19] S. Li and J. Cao, "Distributed adaptive control of pinning synchronization in complex dynamical networks with non-delayed and delayed coupling," *International Journal of Control, Automation and Systems*, vol. 13, no. 5, pp. 1076–1085, 2015.
- [20] G. Chen, "Pinning control and synchronization on complex dynamical networks," *International Journal of Control, Automation and Systems*, vol. 12, no. 2, pp. 221–230, 2014.
- [21] A. Ghaffari and S. Arebi, "Pinning control for synchronization of nonlinear complex dynamical network with suboptimal SDRE controllers," *Nonlinear Dynamics*, vol. 83, no. 1-2, pp. 1003–1013, 2016.
- [22] C. Li and G. Chen, "Synchronization in general complex dynamical networks with coupling delays," *Physica A. Statistical Mechanics and Its Applications*, vol. 343, no. 1–4, pp. 263–278, 2004.
- [23] X.-M. Zhang, M. Wu, J.-H. She, and Y. He, "Delay-dependent stabilization of linear systems with time-varying state and input delays," *Automatica*, vol. 41, no. 8, pp. 1405–1412, 2005.
- [24] Y. Wang, Z. Wang, and J. Liang, "A delay fractioning approach to global synchronization of delayed complex networks with stochastic disturbances," *Physics Letters, Section A: General, Atomic and Solid State Physics*, vol. 372, no. 39, pp. 6066–6073, 2008.
- [25] Z. Fei, H. Gao, and W. X. Zheng, "New synchronization stability of complex networks with an interval time-varying coupling delay," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 56, no. 6, pp. 499–503, 2009.
- [26] D. Gong, H. Zhang, B. Huang, and Z. Ren, "Synchronization criteria and pinning control for complex networks with multiple delays," *Neural Computing and Applications*, vol. 22, no. 1, pp. 151–159, 2013.
- [27] D. Gong, H. Zhang, Z. Wang, and B. Huang, "Novel synchronization analysis for complex networks with hybrid coupling by handling multitude Kronecker product terms," *Neurocomputing*, vol. 82, pp. 14–20, 2012.
- [28] D. Gong, H. Zhang, Z. Wang, and B. Huang, "Pinning synchronization for a general complex networks with multiple time-varying coupling delays," *Neural Processing Letters*, vol. 35, no. 3, pp. 221–231, 2012.
- [29] D. Gong, F. L. Lewis, L. Wang, and K. Xu, "Synchronization for an array of neural networks with hybrid coupling by a novel pinning control strategy," *Neural Networks*, vol. 77, pp. 41–50, 2016.
- [30] D. Gong, J. Liu, and S. Zhao, "Chaotic synchronisation for coupled neural networks based on T-S fuzzy theory," *International Journal of Systems Science*, vol. 46, no. 4, pp. 681–689, 2015.
- [31] C. Yang, Z. Li, R. Cui, and B. Xu, "Neural network-based motion control of an underactuated wheeled inverted pendulum model," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 11, pp. 2004–2016, 2014.
- [32] C. Yang, Y. Jiang, Z. Li, W. He, and C.-Y. Su, "Neural control of bimanual robots with guaranteed global stability and motion precision," *IEEE Transactions on Industrial Informatics*, In Press.
- [33] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, In press.
- [34] C. Yang, K. Huang, H. Cheng, Y. Li, and C. Su, "Haptic identification by ELM-controlled uncertain manipulator," *IEEE Transactions on Systems, Man, and Cybernetics: Systems, In press*.
- [35] W. He and S. S. Ge, "Vibration control of a flexible beam with output constraint," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 8, pp. 5023–5030, 2015.
- [36] W. He, C. Sun, and S. S. Ge, "Top tension control of a flexible marine riser by using integral-barrier lyapunov function," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 497–505, 2015.

- [37] W. He and S. S. Ge, "Cooperative control of a nonuniform gantry crane with constrained tension," *Automatica*, vol. 66, pp. 146–154, 2016.
- [38] W. He, Y. Ouyang, and J. Hong, "Vibration control of a flexible robotic manipulator in the presence of input deadzone," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 1, pp. 48–59, 2017.
- [39] X. Wu and D. Gao, "Fault tolerance control of SOFC systems based on nonlinear model predictive control," *International Journal of Hydrogen Energy*, vol. 42, no. 4, pp. 2288–2308, 2017.