Hindawi
Complexity
Volume 2022, Article ID 1006197, 19 pages
https://doi.org/10.1155/2022/1006197

WILEY | Hindawi

Research Article

HCDRNN-NMPC: A New Approach to Design Nonlinear Model Predictive Control (NMPC) Based on the Hyper Chaotic Diagonal Recurrent Neural Network (HCDRNN)

Samira Johari,[1] **Mahdi Yaghoobi**,[1] **and Hamid R. Kobravi**[2]

[1]*Department of Electrical Engineering Mashhad Branch, Islamic Azad University, Mashhad, Iran*
[2]*Department of Biomedical Engineering Mashhad Branch, Islamic Azad University, Mashhad, Iran*

Correspondence should be addressed to Mahdi Yaghoobi; yaghoobi@mshdiau.ac.ir

Received 1 December 2021; Revised 30 April 2022; Accepted 5 September 2022; Published 18 October 2022

Academic Editor: Sergey Dashkovskiy

In industrial applications, Stewart platform control is especially important. Because of the Stewart platform's inherent delays and high nonlinear behavior, a novel nonlinear model predictive controller (NMPC) and new chaotic neural network model (CNNM) are proposed. Here, a novel NMPC based on hyper chaotic diagonal recurrent neural networks (HCDRNN-NMPC) is proposed, in which, the HCDRNN estimates the future system's outputs. To improve the convergence of the parameters of the HCDRNN to better the system's modeling, the extent of chaos is adjusted using a logistic map in the hidden layer. The proposed scheme uses an improved gradient method to solve the optimization problem in NMPC. The proposed control is used to control six degrees of freedom Stewart parallel robot with hard-nonlinearity, input constraints, and in the presence of uncertainties including external disturbance. High prediction performance, parameters convergence, and local minima avoidance of the neural network are guaranteed. Stability and high tracking performance are the most significant advantages of the proposed scheme.

1. Introduction

Stewart platform is a six-degree-of-freedom parallel robot that was first introduced by Stewart in 1965 and has potential uses in industrial contexts due to its good dynamics performance, high precision, and high rigidity. The control of the Stewart platform is quite challenging due to the nonlinear characteristics of dynamic parameters and time-varying delays. Stewart platform has more physical constraints than the serial manipulators, therefore solving their kinematics and dynamics problem is more difficult, and developing an accurate model of the Stewart platform has always been a concern for researchers in this field [1].

There has been a lot of research done on using the neural networks to the model nonlinear systems [2–4]. In the study of Chen et al. [5], to control nonlinear teleoperation manipulators, an RBF-neural network-based adaptive robust control is developed. As a result, the RBF neural network is used to estimate the nonlinearities and model uncertainty in system dynamics with external disturbances. To handle parameter variations, the adaptive law is developed by adapting the parameters of the RBF neural network online while the nonlinear robust term is developed to deal with estimation errors. Lu employed a NN approximator to estimate uncertain parametric and unknown functions in a robotic system in the study by Lu and Liu [6], which was subsequently used to construct an adaptive NN controller for uncertain n-joint robotic systems with time-varying state constraints. As outlined in [7], an adaptive global sliding mode controller with two hidden layers is developed. The system nonlinearities were estimated using a new RNN with two hidden layers. An adaptive sliding mode control scheme based on RBFNN-based estimation of environmental parameters on the slave side is proposed in the study by Chen et al. [8] for a multilateral telerobotic system with master-slave manipulators. The environment force is modeled generally.

Changes in the structure of the neural network during the training, as well as the use of chaos theory in the neural network, have been considered to cover the behavioral diversity of nonlinear systems. In the study by Chen and Han and Qiao [9, 10], the number of neurons in the hidden layer is changed online. In the study by Han et al. [11], to optimize the NN structure, a penalty-reward method is used. Aihara presented a chaotic NN model in the study by Aihara et al. [12]. Hopfield NN is introduced in the study of Li et al. and Farjami et al. [13, 14] as a chaotic RNN with a chaotic dynamic established temporarily for searching. Reference [15] introduces a context layer that uses chaotic mappings to produce chaotic behavior in NN throughout the training phase in order to prevent local minima. Reference [16] discusses the designing of a chaotic NN by using chaotic neurons which show chaotic behavior in some of their activity areas. In this aspect, the behavior of the neurons and network will change according to the changes in the bifurcation parameters of the neurons which have mostly been inspired by biological studies. A logistic map is utilized as an activation function in the study by Taherkhani et al. [17], which iteratively generates chaotic behavior in the nodes.

In the study of Dongsu and Hongbin [18], an adaptive sliding controller has been used to identify fixed unknown parameters, followed by external disturbances compensation. In the study of Ghorbani and Sani [19], a Fuzzy NMPC is introduced to handle uncertainties and external disturbances. In the study of Jin et al. [20], different parallel robots' NN-based controlling approaches have been reviewed. The applicability of RNN, feedforward NNs, or both for controlling parallel robots has been discussed in detail, comparing them in terms of controlling efficiency and complexity of calculations.

In this paper, due to the inherent delays of the Stewart platform and the design of the controller based on future changes, special attention is paid to the model predictive control. To predict the system behavior over a predefined prediction horizon, MPC approaches require a precise linear model of the under-control system. Stewart platform is inherently nonlinear and linear models are mostly inaccurate in dynamical nonlinear systems modeling. These all bring up the motivation for using nonlinear models in MPC, leading to NMPC.

The most significant features of NMPCs include the following: (I) nonlinear model utilization, (II) state and input constraints consideration, (III) online minimization of appointed performance criteria, (IV) necessity of solving an online optimal control problem, (V) requirement of the system state measuring or estimation, for providing the prediction. Among universal nonlinear models, which are used for predicting the behavior of the system in future, the neural networks are significantly attractive [21, 22].

The effectiveness of the NNs in nonlinear system identification has increased the popularity of NN-based predictive controllers. Nikdel [23], has presented a NN-based MPC to control a shape memory alloy-based manipulator. For nonlinear system modeling and predictive control, a multiinput multioutput radial basis function neural network (RBFNN) was employed in the study of Peng et al. [24]. The recurrent neural networks (RNN) perform well in terms of modeling dynamical systems even in noisy situations because they naturally incorporate dynamic aspects in the form of storing dynamic response of the system through tapping delay, the RNN is utilized in NMPC in the study of Pan and Wang [25], and the results show that the approach converges quickly. In the study of Seyab and Cao [26], a continuous-time RNN is utilized for the NMPC, demonstrating the method's appropriate performance under various operational settings.

In this paper, we will continue this research using the hierarchical structure of the chaotic RNNs, application to NMPC of a complex parallel robot. This paper's contributions and significant innovations are as follows: (I) a new NMPC based on hierarchical HCDRNNs is suggested to model and regulate typical nonlinear systems with complex dynamics. (II) To overcome the modeling issues of complex nonlinear systems with hard nonlinearities, in the proposed controller, the future output of the under-control system is approximated using a proposed novel hierarchical HCDRNN. Note that the equations of motion of such systems are very difficult to solve by mathematical methods and bring forth flaws such as inaccuracy and computational expenses. (III) The weight updating laws are modified based on the proposed HCDRNN scheme, considering the rules introduced in the study of Wang et al. [15]. (IV) On the one hand, propounding the novel hierarchical structure, and on the other hand, the use of chaos in weights updating rules, significantly reduced the cumulative error. (V) The extent of chaos is regulated based on the modeling error in the proposed HCDRNN, in order to increase the accuracy of modeling and prediction. (VI) The control and prediction horizons are specified based on closed-loop control features. (VII) Weights convergence of the proposed HCDRNN is demonstrated and system stability is assured in terms of the Lyapunov second law, taking into account input/output limitations. Furthermore, the proposed controller's performance in the presence of external disturbance is evaluated.

The remainder of this work is structured as follows: Section 2 describes the suggested control strategy in detail, Section 3 discusses the simulation results to validate the efficiency of the proposed method, and Section 4 discusses the final conclusions.

2. The Proposed Control Strategy

The MPC is made up of three major components: the predictive model, the cost function, and the optimization method. The predictive model forecasts the system's future behavior. Based on the optimization of the cost function and the projected behavior of the system, MPC applies an appropriate control input to the process. This paper uses a novel HCDRNN as the predictive model. Moreover, to optimize the cost function, it uses a type of improved gradient method which utilizes the data predicted by the proposed HCDRNN.

Figure 1 shows a block diagram of the designed control system in which $R(t)$ represents the desired trajectory for the coordinates origin of the moving plane. In which, $y(t)$ and $u(t)$ are the outputs and inputs of the Stewart platform,

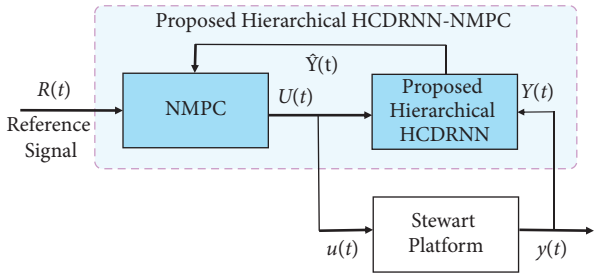


Figure 1: Schematic of the proposed HCDRNN-NMPC.

$\hat{y}(t+1)$ shows the output predicted by the NN model. Finally, the optimization block extracts the control signal, $u(t)$, by minimizing the cost function using the improved gradient descent method.

2.1. Stewart Platform. Figure 2 shows the Stewart platform. All parameters and variables are the same as what Tsai used in [27].

The dynamic model of Stewart platform is introduced in equation (1), which is obtained based on the virtual-work principle [27].

$$-F_z - J_p^{-T}\left(F_p + J_x^T F_x + J_y^T F\right)_y = \overline{\tau}, \tag{1}$$

where $J_p, J_x,$ andJ_y are the manipulator Jacobian matrices, F_p is the resultant of the applied and inertia wrenches exerted at the center of mass of the moving platform, $\overline{\tau}, F_x, F_y,$ andF_z are the vectors of input torque and forces, which are applied to the center of mass of the moving plate from the prismatic joints of the robot. For more details about the robot and its mathematical model, the interested reader can see the reference [27].

2.2. The Proposed Hyper Chaotic Diagonal Recurrent Neural Network. In general, the structure of the NNs may be categorized into feedforward or recurrent types. Possessing the features of having attractor dynamics and data storage capability, the RNNs are more appropriate for modeling dynamical systems than the feedforward ones [28]. Reference [15] introduces the essential concepts of the chaotic diagonal recurrent neural network (CDRNN). This study introduces an HCDRNN, the structure of which is depicted in Figure 3.

The proposed HCDRNN is made up of four layers: input, context, hidden, and output. The hidden layer outputs with v-step delays are routed into the context layer through a chaotic logistic map. The following equations describe the dynamics of the HCDRNN.

$$\begin{aligned}
&\hat{y}(t) = W^o(t)\gamma(t),\\
&\gamma(t) = F(S(t)),\\
&S(t) = W^I(t)X(t) + \begin{bmatrix} \left(W_1^D(t) - a.Z_1(t)\right)\Gamma_1(t-1) \\ \vdots \\ \left(W_n^D(t) - a.Z_n(t)\right)\Gamma_n(t-1) \end{bmatrix},\\
&Z(t+1)\zeta Z(t)\ (1 - Z(t)),
\end{aligned} \tag{2}$$

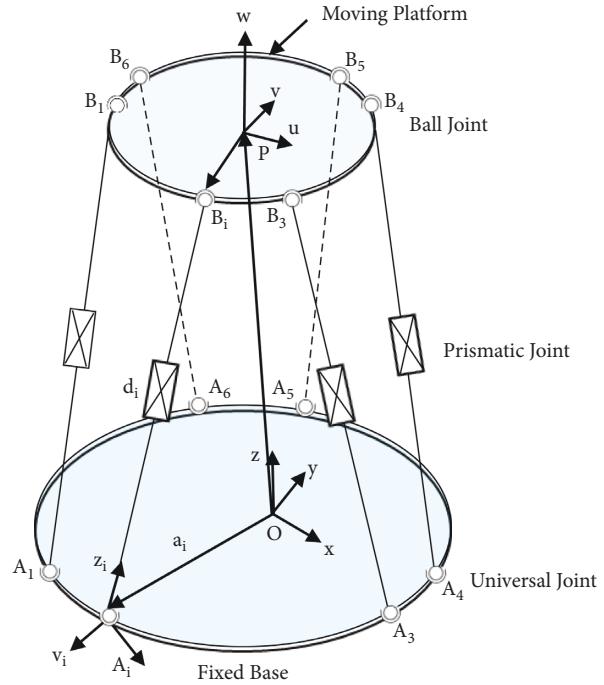


Figure 2: Schematic of Stewart platform [27].

where $X(t) \in R^{m\times 1}$ and $\hat{y}(t) \in R^{1\times 1}$ show the inputs and output of the HCDRNN, $\gamma(t) = [\gamma_1(t), \gamma_i(t), \ldots, \gamma_n(t)]^T \in R^{n\times 1}$ represents the hidden layer's output. A set of $\Gamma_i(t-1) = [\gamma_i(t-1)\ldots\gamma_i(t-\mathrm{v})]^T \in R^{v\times 1}$ is defined as vectors of previous steps' values of $\gamma_i$for$i = 1, 2, \ldots, n$. $F(.)$ shows a symmetric sigmoid activation function. $Z(t) \in R^{n\times 1}$ represents the chaotic logistic map, with $Z(0)$ as a positive random number with normal distribution. The input, context, and output weight matrices are represented as $W^I(t) \in R^{n\times m}$, $W_i^D(t) \in R^{1\times v}(\forall i = 1, 2, \ldots, n)$, and $W^o(t) \in R^{1\times n}$, respectively. $\zeta \in R^{1\times 1}$ is the chaos gain coefficient. The degree of chaos within the HCDRNN can be adjusted by adjusting the parameter a, which ranges from 0 for a simple DRNN to close to 4 for an HCDRNN. This fact allows you to regulate the level of chaos within the NN by altering the parameter a in such a manner that the reduction in training error leads to a progressive decrease in the extent of chaos until it reaches stability. The value of the parameter a's value could be altered as follows. As the change is exponential, the NN will rapidly converge.

$$\begin{aligned}
&\beta(t) = -\frac{\overline{\mathrm{e}}(t)}{\overline{\mathrm{e}}(t)} + 1,\\
&a = \begin{cases} \mu_0 + (\mu_{\max} - \mu_0)\exp\left(\dfrac{-\beta(t)}{T_a}\right), & \overline{\mathrm{e}}(t) > \varepsilon, \\ 0, & \overline{\mathrm{e}}(t) \le \varepsilon. \end{cases}
\end{aligned} \tag{3}$$

where $\overline{\mathrm{e}}(t)$ is the samples' absolute training prediction error. The prediction error, $e_p(t)$, represents the difference between the system's actual output, $y(t)$, and the output of HCDRNN, $\hat{y}(t)$.

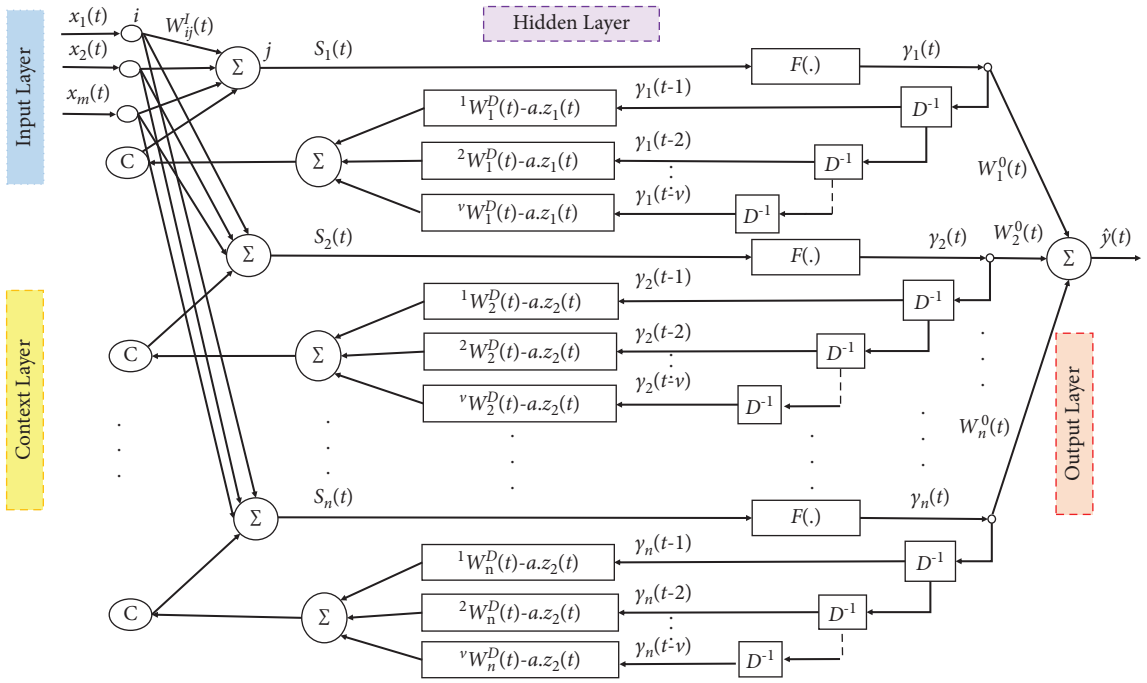


Figure 3: Structure of the HCDRNN [29].

$$e_p(t) = y(t) - \hat{y}(t). \tag{4}$$

μ_{max} and μ_o represent the maximum and minimum threshold of the parameter a, respectively. T_a is the annealing parameter, and ε is the prediction error threshold. To minimize the error function, $E_p(t)$, the weight update laws for the output, hidden, and context layers are based on the robust adaptive dead zone learning algorithm reported in [30].

$$E_p(t) = \frac{1}{2}e_p^2(t). \tag{5}$$

Accordingly, weights updating laws are modified here for the proposed structure of the HCDRNN as follows [29]:

2.2.1. Output Layer. If $e_p(t) < \Delta^{\mathbf{o}}(t)$ then $W^o(t+1)$ and $\Delta^{\mathbf{o}}(t+1)$ do not change, otherwise:

$$W^o(t+1) = W^o(t) + \frac{2e_p(t)\gamma(t)^T}{1-\gamma(t)^2}, \tag{6}$$

$$\Delta^{\mathbf{o}}(t+1) = \Delta^{\mathbf{o}}(t) + \frac{2\bar{e}(t)}{1-\gamma(t)^2}. \tag{7}$$

2.2.2. Hidden Layer. If $e_p(t) < \Delta^{\mathbf{I}}(t)$ then $W^I(t+1)$ and $\Delta^{\mathbf{I}}(t+1)$ do not change, otherwise:

$$W^I(t+1) = W^I(t) + \frac{2(t)e_p(t)W^o(t)F'(t)X(t)}{1-W^o(t)F'(t)X(t)^2}, \tag{8}$$

$$\Delta^I(t+1) = \Delta^I(t) + \frac{2F'_{\min}(t)\bar{e}(t)}{1-W^o(t)F'(t)X(t)^2}, \tag{9}$$

where $F'(t) = (1-\gamma^2(t))$ is the first derivative of the activation function in the hidden layer and $F'_j(t) = F'(s_j(t)), \forall i = 1, 2, \ldots, n$, and $F'_{\min}(t) = \min(F'_j(t)) \neq 0, \forall j, t$.

2.2.3. Context Layer. If $e_p(t) < \Delta^{\mathbf{D}}(t)$ then $W^D(t+1)$ and $\Delta^D(t+1)$ do not change, otherwise $\forall i = 1, 2, \ldots, n$:

$$W_i^D(t+1) = W_i^D(t) + \frac{2F'_{\min}(t)e_p(t)W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}, \tag{10}$$

$$\Delta^D(t+1) = \Delta^D(t) + \frac{2F'_{\min}(t)\bar{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}. \tag{11}$$

In these equations, "$\Delta^o(t), \Delta^I(t), \Delta^D(t)$ are the robust adaptive dead zones for output, hidden and context layers, respectively."

Remark 1. Theorems A.1, A.2, and A.3 in the appendix prove the convergence of neural network weights.

Remark 2. As illustrated below [11], it is expected that a multiinput single-output nonlinear autoregressive exogenous (NARX) model may represent the undercontrol nonlinear system utilizing the n_u delayed system's inputs and n_y delayed system's outputs.

$$y(t) = g\left(y(t-1), \ldots, y\left(t-n_y\right), u(t-1), \ldots, u\left(t-n_u\right)\right). \tag{12}$$

In this equation, $g(.)$ is an unknown function.

Based on Remark 1 and Remark 2, an array of H_p HCDRNNs is used to forecast the system's behavior in a H_p-step-ahead prediction horizon after the training and weights updating operations. The structure of this HCDRNN array is depicted in Figure 4.

Remark 3. Each HCDRNN in the array, as shown in Figure 4, is trained independently, and its weight matrices differ from those of the other HCDRNNs. As a result, the formulation in Sections 2.2.1 to 2.2.3 should be changed based on the input-output permutation for each element of the hierarchy. Remark 1 is, however, applied to all of the HCDRNNs in the array.

2.3. The Proposed HCDRNN-NMPC. A finite-horizon NMPC cost function would be the same as indicated in reference [11].

$$\widehat{V}(t) = \rho_1 [R(t) - \widehat{Y}(t)]^T [R(t) - \widehat{Y}(t)] + \rho_2 \Delta U(t)^T \Delta U(t), \tag{13}$$

where $R(t) = [r(t+1), r(t+2), \ldots, r(t+H_p)]^T$ is the reference signal, $Y(t) = [y(t+1), y(t+2), \ldots, y(t+H_p)]^T$ is the system output, and $\widehat{Y}(t) = [\widehat{y}(t+1), \widehat{y}(t+2), \ldots, \widehat{y}(t+H_p)]^T$ is the predicted output through the prediction horizon. $\Delta U(t) = [\Delta u(t), \Delta u(t+1), \ldots, \Delta u(t+H_u-1)]^T$ is the control signal variations during the upcoming control horizon. ρ_1 and ρ_2 are weighting parameters, determining the significance of the tracking error versus the control signal variation in the cost function, $\widehat{V}$. H_p is prediction horizon and H_u is control horizon ($H_u < H_p$). However, the equation faces the following constraints [11]:

$$\begin{aligned} &|\Delta u(t)| \le \Delta u_{\max}, u_{\min} \le u(t) \le u_{\max}, \\ &\widehat{y}_{\min} \le \widehat{y}(t) \le \widehat{y}_{\max}, \\ &r\left(t + H_p + i\right) - \widehat{y}\left(t + H_p + i\right) = 0, \forall \mathrm{i} \ge 1. \end{aligned} \tag{14}$$

The control signal, $U(t+1) = [u(t+1), u(t+2), \ldots, u(t+H_u)]^T$, based on the improved gradient method is given below [11, 31]:

$$U(t+1) = U(t) + \Delta U(t) = U(t) - \eta \frac{\partial \widehat{V}(t)}{\partial U(t)}, \tag{15}$$

$$\Delta U(t) = \frac{\eta \rho_1}{1 + \eta \rho_2} \left(\frac{\partial \widehat{Y}(t)}{\partial U(t)} \right)^T (R(t) - \widehat{Y}(t)), \tag{16}$$

where $\eta > 0$ represents the learning rate of the control input sequence and $\partial \widehat{Y}(t)/\partial U(t)$ represents the Jacobian matrix, J, which is computed as a matrix with the dimension of $H_p \times H_u$.

$$\frac{\partial \widehat{Y}(t)}{\partial U(t)} = \begin{bmatrix} \frac{\partial \widehat{y}(t+1)}{\partial u(t)} & 0 & 0 & \cdots & 0 \\ \frac{\partial \widehat{y}(t+2)}{\partial u(t)} & \frac{\partial \widehat{y}(t+2)}{\partial u(t+1)} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{\partial \widehat{y}(t+H_u)}{\partial u(t)} & \frac{\partial \widehat{y}(t+H_u)}{\partial u(t+1)} & \frac{\partial \widehat{y}(t+H_u)}{\partial u(t+2)} & \cdots & \frac{\partial \widehat{y}(t+H_p)}{\partial u(t+H_u-1)} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial \widehat{y}(t+H_p)}{\partial u(t)} & \frac{\partial \widehat{y}(t+H_p)}{\partial u(t+1)} & \frac{\partial \widehat{y}(t+H_p)}{\partial u(t+2)} & \cdots & \frac{\partial \widehat{y}(t+H_p)}{\partial u(t+H_u-1)} \end{bmatrix}_{H_p \times H_u}. \tag{17}$$

The Jacobian matrix, $J = \partial \widehat{Y}(t)/\partial U(t)$, can be computed based on the chain rule.

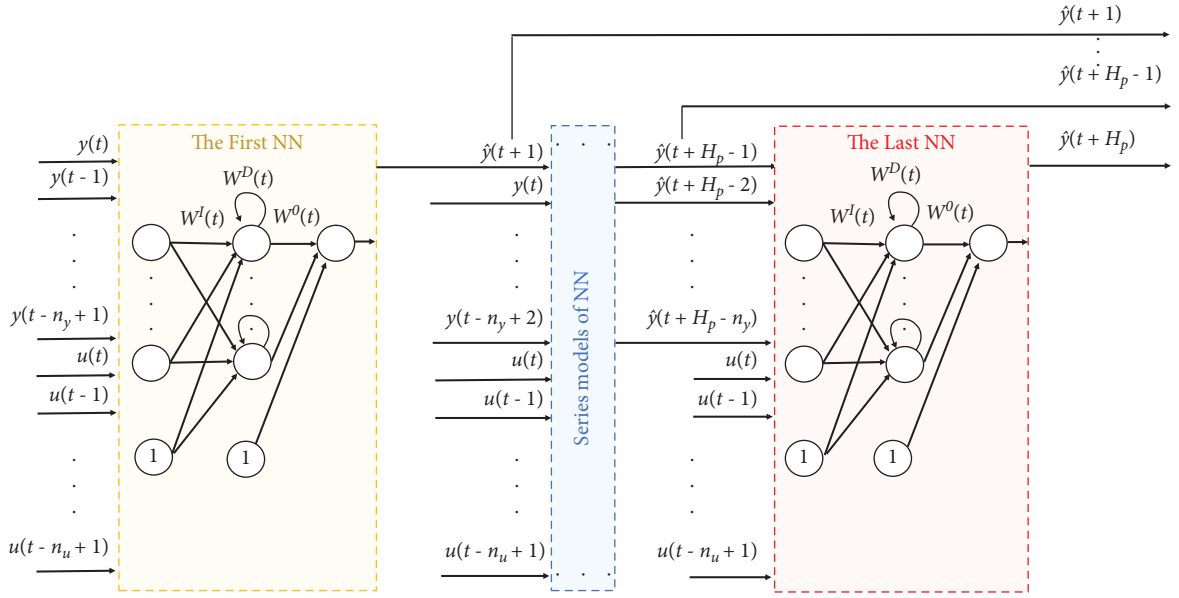


Figure 4: Prediction of H_p-step-ahead outputs by the hierarchical HCDRNN [29].

$$\frac{\partial \hat{y}(t+i)}{\partial u(t+j)} = W^{o}(t+i)\frac{\partial \gamma(t+i)}{\partial u(t+j)}, \forall i = 1, 2, \ldots, H_p, \forall j = 0, 1, \ldots, (H_u - 1),$$

$$\frac{\partial \hat{y}(t+i)}{\partial u(t+j)} = F'(t+i)\left(W^{I}(t+i)\frac{\partial X(t+i)}{\partial u(t+j)} + \begin{bmatrix} \left(W_1^D(t+i) - a.Z_1(t+i)\right)\dfrac{\partial \Gamma_1(t+i-1)}{\partial u(t+j)} \\ \vdots \\ \left(W_n^D(t+i) - a.Z_n(t+i)\right)\dfrac{\partial \Gamma_n(t+i-1)}{\partial u(t+j)} \end{bmatrix}\right), \tag{18}$$

in which, $F'(t+i) = \partial F(S(t+i))/\partial S(t+j) = (1-\gamma^2(t))$ and $\partial \gamma(t+i)/\partial u(t+j)$ can be computed recurrently knowing that if $j = i$ then $\partial \gamma(t+i)/\partial u(t+j) = 0$. It means that the computations should be completed from $\partial \gamma(t+i)/\partial u(t+i-1)$ to $\partial \gamma(t+i)/\partial u(t+j)$. Moreover, considering the structure of $X(t+i)$, the $\partial X(t+i)/\partial u(t+j)$ can be calculated recurrently based on equation (26).

Algorithm 1 summarizes the proposed HCDRNN-NMPC scheme [29].

In steps 4 and 5 of the proposed algorithm, as the number of system inputs increases, the dimensions of the Jacobian matrix increase, and the estimation error increases due to the discretizations performed in calculating the derivatives. Choosing the appropriate sampling, time is used as a solution to reduce the estimation error in this paper.

2.3.1. Stability Analysis for HCDRNN-NMPC. The stability of NMPC-HCDRNN is demonstrated by considering the convergence of the model, which is proved in Remark 1 and Appendix, and the fact that the neural network training is done offline.

Theorem 1. *Consider the constrained finite-horizon optimal control presented by (17) and (18). Lyapunov's second law ensures the asymptotic stability of the proposed controller due to the limited input and output amplitudes and the semi-definite negative $\hat{V}(t)$ if the neural network weights' convergence is proven and the predictive control law is as given in equations (19) and (20).*

Proof. The constrained finite-horizon optimal control given in equation (13) can be rewritten as in (19) by rewriting the cost function along the control horizon:

$$\hat{V}(t) = \rho_1 \sum_{i=1}^{H_p} r\left[(t+i) - \hat{y}(t+i)\right]^2 + \rho_2 \sum_{i=1}^{H_u} \Delta u^2(t+i-1). \tag{19}$$

$U(t) = [u(t), u(t+1), \ldots, u(t+H_u-1)]^T$ is the optimal control sequence obtained at time t using the optimization algorithm. If $U_s(t+1)$ is the suboptimal control sequence extracted from $U(t)$ and considered as $U_s(t+1) = [u(t+1), \ldots, u(t+H_u-1)]^T$, the suboptimal cost function $\hat{V}_s(t+1)$ is defined as follows:

Step 1. Determine H_p and H_u, such that $H_p > H_u$.
Step 2. Get $R(t)$, $U(t)$, and $X(t)$ in each control step, such that:
(i) $R(t) = [r(t+1), \ldots, r(t+H_p)]^T$ is the desired values for H_p next steps.
(ii) $U(t) = [u(t), \ldots, u(t+H_u-1)]^T$ is the last optimal sequence of the predicted control signal.
(iii) $X(t)$ is the delayed input-output vector of nonlinear system.
Step 3. Predict the outputs of the system for H_p next steps by the proposed HCDRNN.
Step 4. Calculate J by equation (17).
Step 5. Compute $\Delta U(t)$ and $U(t+1)$ by equations (15) and (16), respectively.
Step 6. Apply $U(t+1)$ as the first element of vector $U(t+1)$ to the nonlinear system, and go back to Step 2 for the next sample time.

Algorithm 1: Details of HCDRNN-NMPC scheme.

$$\widehat{V}(t+1) = \rho_1 \sum_{i=1}^{H_p+1} [r(t+i) - \widehat{y}(t+i)]^2 + \rho_2 \sum_{i=1}^{H_u} \Delta u^2(t+i-1). \tag{20}$$

Using the difference of $\widehat{V}(t)$ and $\widehat{V}_s(t+1)$, and assuming that $e(t+i) = r(t+i) - \widehat{y}(t+i)$, equation (21) is written as follows:

$$\begin{aligned} \widehat{V}_s(t+1) - \widehat{V}(t) &= \left(\rho_1 \sum_{i=1}^{H_p+1} e^2(t+i) + \rho_2 \sum_{i=1}^{H_u} \Delta u^2(t+i-1) \right) - \left(\rho_1 \sum_{i=1}^{H_p} e^2(t+i) + \rho_2 \sum_{i=1}^{H_u} \Delta u^2(t+i-1) \right), \\ &= \rho_1 \left(e^2\left(t+H_p+1\right) - e^2(t+1) \right) - \rho_2 \Delta u^2(t) \le 0. \end{aligned} \tag{21}$$

Therefore, if $U(t+1)$ is the optimal solution of the optimization problem time $(t+1)$ using the control law described in equation (16), it outperforms $U_s(t+1)$, which is suboptimal and its cost function is smaller according to equation (22).

$$\widehat{V}(t+1) - V(t) \le \widehat{V}_s(t+1) - \widehat{V}(t) \le 0. \tag{22}$$

Hence, the proof is complete. □

3. Simulation

To control the Stewart platform, HCDRNN-NMPC is used such that the upper moving plane of the platform tracks the desired trajectory. The simulations have been carried out by MATLAB software, 2015 version. To evaluate the efficiency of the control method against external disturbances, the effects of the disturbance applied to the force on one of the links of Stewart platforms have been investigated.

3.1. Neural Network-Based Model. To predict the behavior of the Stewart platform, input-output data of the system under different operating configurations are required. To generate the training data, the inverse dynamics of the Stewart platform are solved for several random desired trajectories, based on the algorithm presented by [27] and the parameters introduced by [32]. The applied sampling time is $t_s = 0.01$.

3.1.1. Training. The general structure of the HCDRNN is designed in such a way that its inputs vector, $X(t)$, includes the previous position of the moving plane, $p_x(t)$, and the forces exerted on each link, $F_i(t)$, as in equation (23), and its output vector, $y(t)$, includes the position of the moving plane as in equation (24).

$$X(t) = \left[F_1(t), \ldots, F_1\left(t-n_u\right), \ldots, F_6(t), \ldots, F_6\left(t-n_u\right), p_x(t-1), \ldots, p_x\left(t-n_y\right) \right], \tag{23}$$

$$y(t) = P_x(t). \tag{24}$$

The number of elements of $X(t)$ determines the number of input layer neurons. Accordingly, thirteen input nodes have been considered for six links. As the network's output comprises three positions and three directions, separate networks should be considered for each output and, therefore, there would be six MISO networks in our case. For the aforementioned networks, a supervised learning scheme is considered to train the networks with regard to the inputs and outputs. Divided into two sets, 70% of the data were chosen for training and 30% of them for testing. At the beginning of the training of the HCDRNNs, the weight matrices are randomly valued. Tangent sigmoid is selected as the activation function, and the input-output data are normalized. The values for the NN parameters are defined as

$\varepsilon = 0.01$, $T_a = 0.07$, $\mu_{\max} = 4$, $\mu_0 = 0$, $\mu_o = \mu_I = \mu_D = 0.01$, $n_y = 2$, and $n_u = 3$ as well as $\mathbf{W^I}, \mathbf{W^D}, \mathbf{W^o}, \Delta^{\mathrm{I}}, \Delta^D, \Delta^o$ are randomly initialized.

The neural network training is done offline, but during the training, the coefficient a is adjusted online in such a way that the behavior of the Stewart platform is covered by creating chaos in the neural network structure, and as the training error decreases, its value changes such that the neural network's chaos is decreased. Figure 5 shows the chaotic property of the HCDRNN.

The impact of the number of hidden layer neurons on the approximation performance is studied. Table 1 reports the results of this study for 7 to 43 neurons, where their performances are compared in terms of training time and MSE.

As shown in Table 1, the training time increases with an increase in the number of neurons in the hidden layer. Considering the fact that the MSE value is proper when there are 7 neurons in the hidden layer, it would be more appropriate to use 7 neurons in the hidden layer for the prediction of the Stewart platform.

For a sinusoidal trajectory, the results of one-step-ahead, two-step-ahead, and three-step-ahead system behavior predictions are investigated. Table 2 reports the MSE of the prediction error.

Table 2 indicates a reliable prediction by the HCDRNN without any accumulated error. As a significant conclusion, it is shown that the use of chaotic context layer, besides the use of different weight matrices that were trained for each step, in the proposed hierarchical structure, overcome the error accumulation in n-step-ahead predictions.

3.2. The Results of HCDRNN-NMPC. In this paper, the values for the parameters are considered as follows: $H_p = 3$, $H_u = 2$, $\rho_1 = 0.8$, $\rho_2 = 0.2$, and $\eta = 0.01$. The performance of the NMPC is compared with the MPC, which both are evaluated by the integral absolute error (IAE).

$$\mathrm{IAE} = \frac{1}{T}\sum_{t=1}^{T} |R(t) - Y(t)|, \tag{25}$$

where T shows the total number of samples. Some research studies are using other metrics like mean square error (MSE) and/or integral square error (ISE). Each of these metrics has drawbacks that led us to use of IAE instead. Metrics that use error squares magnify the errors greater than one and minify the errors less than one, which is not precise in robotics motion errors. Input and output signals are bounded in the intervals mentioned in equation (26).

$$0 \le F_i(t) \le 16, -1.7 \le P_x(t) \le 1.2. \tag{26}$$

3.2.1. Sample Trajectories. The performance of the controller to track the three paths demonstrated in sections (1), (2), and (3).

(1) A sample trajectory has been designed, which is intended to be tracked with the undercontrol Stewart platform. Trying to calculate the control signal applied to link 1, assuming that the forces exerted on other links remain fixed.

$$P(t) = \begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix} = \begin{bmatrix} -1.5 + 0.2\sin(\omega t) \\ 0.2\sin(\omega t) \\ 1.0 + 0.2\sin(\omega t) \end{bmatrix}, \tag{27}$$

where $\omega = 3$ and $0 \le \omega t \le 2\pi$. Considering the desired trajectory for the robot's movement and the actual trajectory on x, y, z axis, tracking error varies within the range of $[-5.625e-3, 0.015e-3]$ for all three axes which are neglectable.

Figure 6 shows the three-dimensional path of the top plane of the Stewart platform. As found in Figure 6, the NMPC has extracted the control signal in a way that the Stewart platform's output tracks the reference signal along three axes well. Moreover, Figure 7 shows the force exerted to link 1.

As it is shown in Figure 7, the control signal applied to link 1 provides for the limit needed for the forces exerted in each link.

(2) The second sample trajectory sets out to track the following two-frequency trajectory.

$$\phi_x = \phi_y = \phi_z = 0,$$

$$P(t) = \begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix} = \begin{bmatrix} 0.2\sin\omega_1 t + 0.4\sin\omega_2 t \\ 0.2\sin\omega_1 t \\ 1.0 + 0.2\sin\omega_2 t \end{bmatrix} m, \omega_1 = 3, \omega_2 = 2, 0 \le \omega_1 t, \omega_2 t \le 2. \tag{28}$$

Considering the desired trajectory for the robot's movement and the actual trajectory on x, y, z axis, tracking error range on x, y, z axis, is reported in Table 3.

Taking Table 3 into consideration, it can be concluded that the tracking error varies within the range of $[-1.01\_2.03] * e - 8$ for all three axes which are neglectable.

Figure 8 shows the three-dimensional path of the top plane of the Stewart platform, and Figure 9 shows the force exerted to link 1.

As found in Figure 9, the NMPC has extracted the control signal in a way that the Stewart platform's output tracks the reference signal along three axes well. As found in Figure 9, the control signal applied to link 1 provides for the limit needed for the forces exerted in each link, assuming that the positions of other links remain unchanged.

(3) In the third sample path, which is similar to the path presented in the reference [33], to evaluate the performance of the proposed controller in tracking the paths with rapid changes, the following two-level step path has been examined.

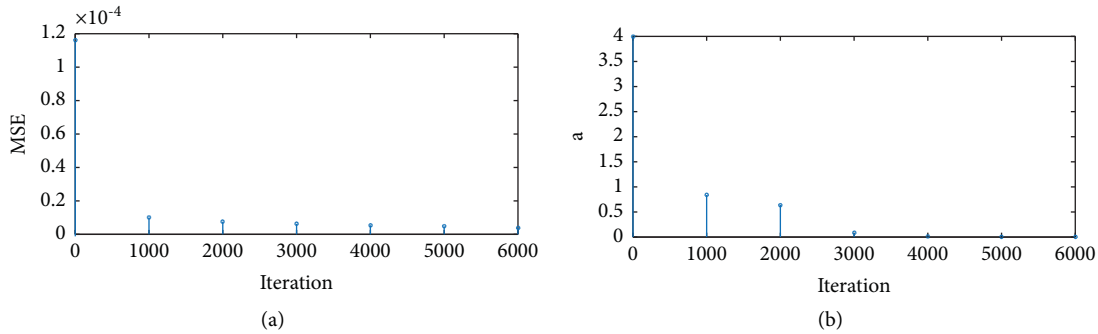


Figure 5: Traces of the training (a) traces of the training mean square error (MSE) and (b) traces of the coefficient a.

Table 1: Comparison of prediction performance versus the number of hidden layer neurons.

No. of hidden layer neurons	Training MSE	Training time (sec)
7	$9.7927e-5$	658.28
27	$9.6511e-8$	978.12
43	$9.4508e-5$	1804.05

$$\phi_x = \phi_y = \phi_z = 0,$$

$$P(t) = \begin{bmatrix} P_x(t) \\ P_y(t) \\ P_z(t) \end{bmatrix} = \begin{bmatrix} 3.5(\mathrm{u}(t) - \mathrm{u}(t-5)) \\ 3(\mathrm{u}(t) - \mathrm{u}(t-1)) \\ 5.3(\mathrm{u}(t) - \mathrm{u}(t-7)) \end{bmatrix} cm, \quad (29)$$

$$0 \le P_x(t), P_y(t), P_z(t) \le, 0 \le F_i(t) \le 6.$$

Considering the desired trajectory for the robot's movement and the actual trajectory on x, y, z axis, tracking error range on x, y, z axis is reported in Table 4. The three-dimensional path traveled by the Stewart platform is shown in Figure 10. The force exerted to link 1 is shown in Figure 11.

As shown in Figures 10 and 11, due to intensive changes in the desired path, the controller made a control effort to extract the control signal in order to reach the desired path, and after reaching the desired path, the control signal did not change. The transient phase of the response is well observed in this optimal path, and as reported in Table 4, the tracking error in tracking the reference signal for x, y, and z axes changes between [−5.1–5.5], which is desired considering the severe changes of the reference and the control signal applied to link 1 satisfies the applied force's constraint.

3.3. External Disturbance Rejection. The effect of external disturbance is assessed here, in the form of a pulse signal with $0.4N$ amplitude, during 1 to 1.4 seconds, applying to another link's force. The proposed control performance is shown in Figure 12.

Figure 13 demonstrated the tracking error and control signal in the presence of external disturbance.

By applying disturbance, the output is changed and a control effort is made to enhance the tracking. Some advantage of the proposed method can be the low number of oscillations in disturbance rejection, the smaller overshoot and undershoot than the initial disturbance magnitude, resulting in a more uniform output, and a significant improvement that is the smaller and more smooth control signal.

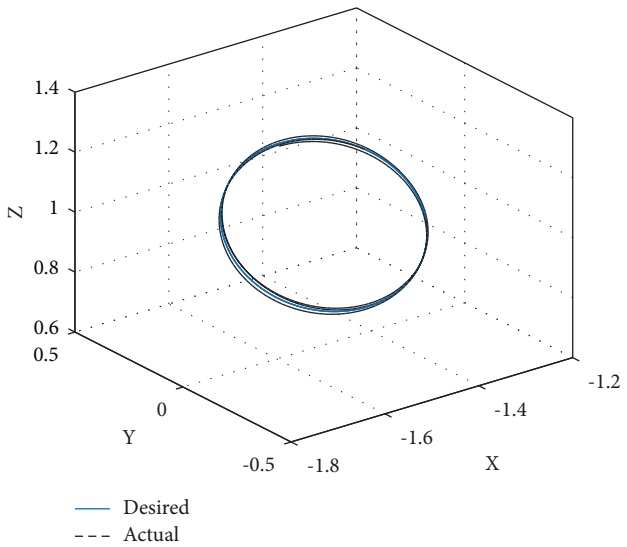


Figure 6: Tracking the three- dimensional path.

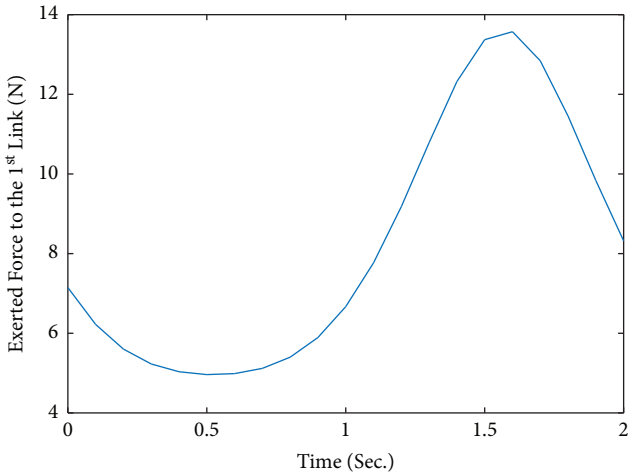


Figure 7: Force exerted to link 1 (control signal).

Table 2: MSE of prediction error for a sinusoidal trajectory.

No. of step ahead	One-step-ahead	Two-step-ahead	Three-step-ahead
MSE of prediction error	$8.9321e-8$	$1.2463e-6$	$9.7927e-5$

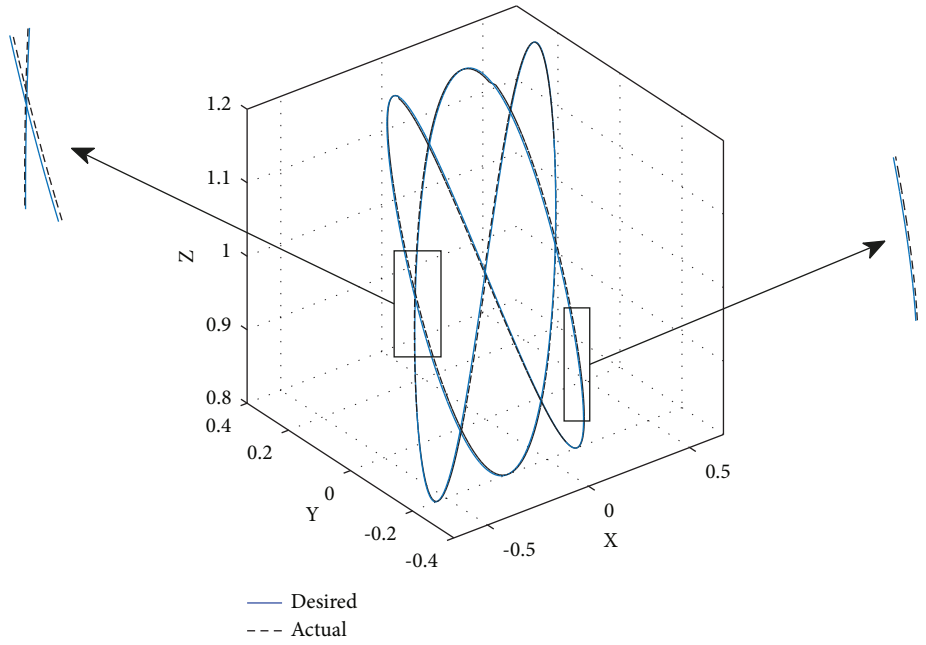


Figure 8: Tracking the three-dimensional path.

Table 3: Tracking error range on x, y, z axis for two-frequency trajectory.

Axis	x	y	z
Tracking error range	$[-1.01\_2.03]*e-8$	$[-1.01\_1.88]*e-6$	$[-0.79\_1.05]*e-5$

3.4. Simulation Results Analysis. The nonlinear model predictive controller requires high computational to extract the control signal, but because of the system's nonlinear model, it can achieve the desired control performance with minimum error. Because the Stewart platform has unknown dynamics, the NN was used to model it. Chaos theory was used in NN to reduce and speed up control calculations, which accelerates the learning dynamics and thus solves the problem of predictive control being slow. Furthermore, involvement with local minimums is avoided by employing chaos in the neural network and increasing the order of chaos by employing more chaotic functions in the hidden layer, resulting in hyper-chaos in the proposed neural network. Table 5 compares the prediction performance of the DRNN and proposed HCDRNN.

Table 6 compares the performance of the proposed control with the proportional-integral-derivative (PID) control [34], the sliding mode control [18], and the fuzzy NMPC [19] and DRNN-NMPC. The comparison results are recorded in terms of IAE.

The proposed method provides a minor value of IAE compared with the other method.

PID controllers need a set of big gains for the proportional, integral, and derivative coefficients, and this makes the control signal highly sensitive to external disturbance so that the control signal rises to a large value with the lowest level of disturbance. However, the control inputs are bounded for factual reasons, thus, the control signal computed through the PID controller would not be applicable in practice.

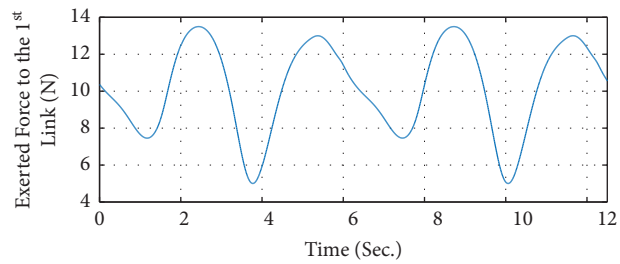


Figure 9: Force exerted to link 1 (control signal).

As demonstrated in Figure 13(a), when the external disturbance is applied to the robot, tracking error has a small value in the range of $[-6e-4, -3e-3]$, which is proof of the proposed method's high performance.

Table 4: Tracking error range on x, y, z axis for two-level step trajectory.

Axis	x	y	z
Tracking error range	[−3.2_3.5]	[−1.8_3]	[−5.1_5.5]

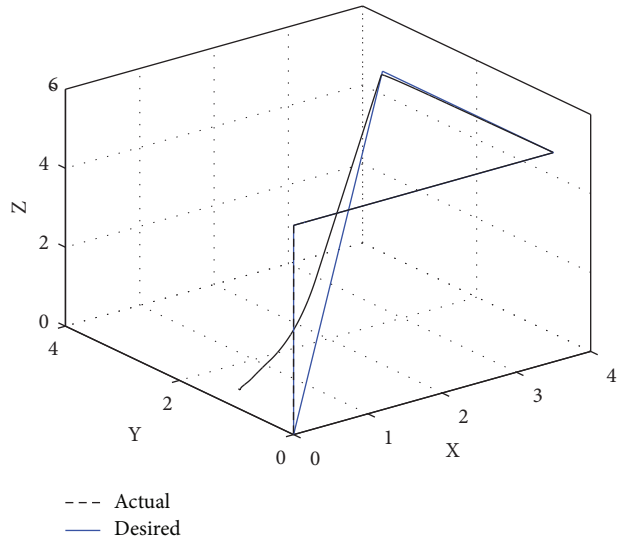


Figure 10: Tracking the three-dimensional path.

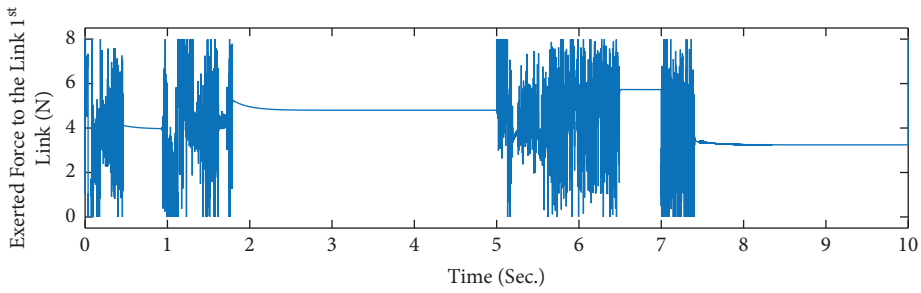


Figure 11: Force exerted to link 1 (control signal).

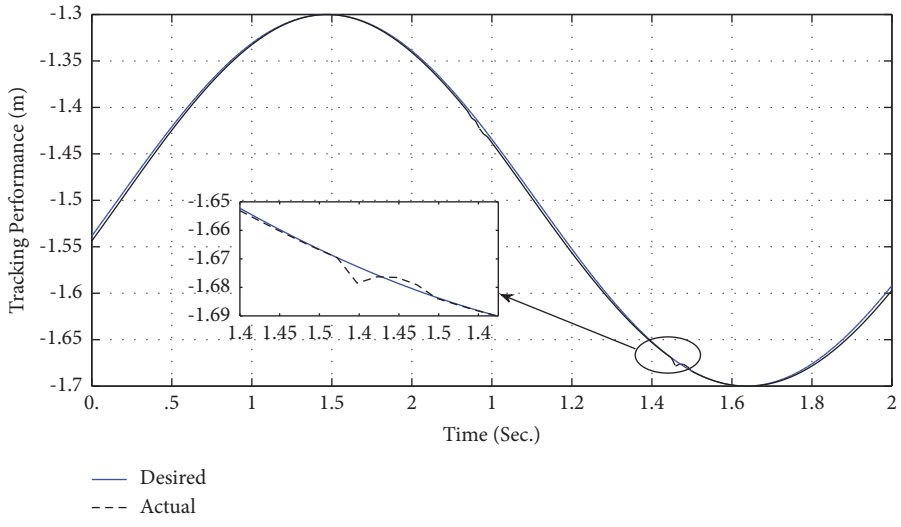
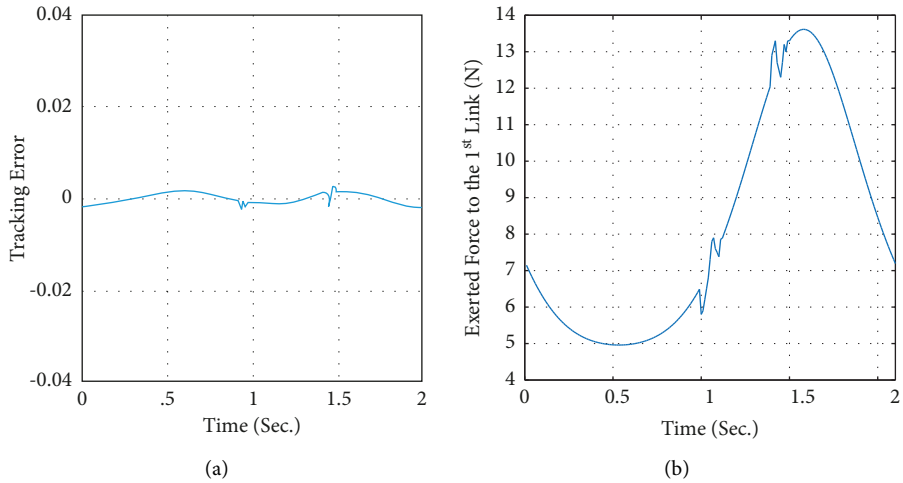


Figure 12: Disturbance rejection.



(a) (b)

Figure 13: (a) Tracking error and (b) extracted control signal in the presence of external disturbance.

Table 5: Prediction performance of the DRNN and proposed HCDRNN.

NN	DRNN		HCDRNN	
Prediction error	Training	Test	Training	Test
Step1	$1.7545e-6$	$4.2148e-5$	$3.8214e-8$	$1.1284e-7$
Step2	$7.2561e-4$	$6.7316e-3$	$9.1852e-8$	$5.6371e-7$
Step3	$9.1027e-1$	$1.4111e-1$	$6.2379e-7$	$9.0141e-7$

Table 6: IAE comparison.

Control approach	PID	Sliding mode control	Fuzzy NMPC	DRNN-NMPC	The proposed method
IAE	$3.2e-4$	$2.8e-5$	$4.6e-6$	$1.9e-3$	$2.9e-8$

4. Conclusion

This paper proposed a novel hierarchical HCDRNN-NMPC for modeling and control of complex nonlinear dynamical systems. Numerical simulations on the control of a Stewart platform are prepared to demonstrate the performance of the proposed strategy in tracking and external disturbance rejection. One of the most essential aspects of the suggested method is its hierarchical HCDRNN's ability to accurately estimate the system's output via a forward-moving window. The hierarchical structure enables the proposed mechanism to precisely adjust each HCDRNN for predicting the outputs of the system for only one specified sample ahead. This enhances the ability of the predictive model in adapting with variations of the complex dynamical systems. This paper provides the adaptive weight update rules for the proposed v-step delayed HCDRNN. Moreover, for determining the sequence of the control signal, an enhanced gradient optimization method is used. Results of the provided simulations and comparisons indicate superior performance of the proposed control system in tracking and removing the effect of the external disturbances. In future research studies, the effects of merging HCDRNNs instead of the hierarchical structure will be investigated, which will last to create a deep HCDRNN structure. The suggested controller's robustness against various forms of disturbances will also be tested.

Appendix

The adaptive dead zone vector method described in [15] was used to demonstrate the convergence of neural network weights for each layer. As a result, the relationship between neural network prediction error, output error, and bounded disturbance of each layer has been used.

A Proof of Weight Convergence for the Output Layer

In order to evaluate the convergence of the weights of the output layer, the predictive error of the neural network, $e_p(t)$, is defined as in the following equation (A.1).

$$\begin{aligned} e_p(t) &= y(t) - \widehat{y}(t) + v(t), \\ &= W^{o*}(t)\gamma^*(t) - W^o(t)\gamma(t) + v^o(t), \\ &= W^{o*}(t)\gamma^*(t) - W^{o*}\gamma(t) + W^{o*}\gamma(t) - W^o(t)\gamma(t) + v^o(t), \end{aligned} \tag{A1}$$

in which $y(t)$ is the desired output, $\hat{y}(t)$ is the output predicted by the neural network, and $v(t)$ is the noise. W^{o*} and γ^* are the optimal or suboptimal values of the output layer's weight and output of the hidden layer of the neural network. Equation (A.2) describes the relationship between the predictive error of the neural network, error of the output layer $e^o(t)$, and bounded disturbance of the output layer $\tilde{v}^o(t)$.

$$\begin{aligned} e_p(t) &= \left(W^{o*}(t)\gamma^*(t) - W^{o*}\gamma(t) + v^o(t)\right) \\ &\quad - \left(W^o(t)\gamma(t) + W^{o*}\gamma(t)\right), \\ e_p(t) &= \tilde{v}^o(t) - e^o(t), \end{aligned} \tag{A2}$$

Theorem A.1. *Assume that the dead zone vector is defined as* $\Delta^o = [\Delta_1^o \ldots \Delta_k^o \ldots \Delta_h^o]^T$ *in which* $\Delta_k^0(t) \geq max(|\tilde{v}_k^o(t)|)$. *If the weights of the neural network and the dead zone vector is updated such that equation (A.3) is satisfied:*

$$\% \lim_{t\longrightarrow\infty} \sup\left(\tilde{W}^o(t+1) - \tilde{W}^o(t) - \left(\tilde{\Delta}^o(t+1) - \tilde{\Delta}^o(t)\right)\right) \leq 0. \tag{A3}$$

Then, the neural network's error is limited as follows and the weight of the output layer converges.

$$\lim_{t\longrightarrow\infty} \sup e_p(t) \leq \Delta^{o*} \tag{A4}$$

$\tilde{W}^o(t) = W^o(t) - W^{o*}$, $\tilde{\Delta}^o(t) = \Delta^o(t) - \Delta^{o*}$, Δ^{o*}, and W^{o*} are the optimal or suboptimal value of the dead zone vector and weight of the output layer.

Proof. By using $\tilde{W}^o(t) = W^o(t) - W^{o*}$ and substituting equation (10) and equation (11) in equation (A.3), we have:

$$= \lim_{t\longrightarrow\infty}\left(\frac{4e_p(t)\gamma(t)\left(W^o(t) - W^{o*}\right)}{1-\gamma(t)^2} + \frac{2e_p(t)\gamma(t)^2}{1-\gamma(t)^2} - \frac{4\bar{e}(t)}{1-\gamma(t)^2}\tilde{\Delta}^o(t) - \frac{2\bar{e}(t)}{1-\gamma(t)^2}^2\right). \tag{A5}$$

By substituting equation (A.2) in equation (A.5), we have:

$$= \lim_{t\longrightarrow\infty}\left(\sum_{k=1}^{h}\frac{4e_{pk}(t)\left(\tilde{v}_k^o(t) - \Delta_k^o(t) - e_{pk}(t)\right)}{1-\gamma(t)^2} + \frac{2e_p(t)\gamma(t)^2}{1-\gamma(t)^2} + \frac{4\bar{e}(t)}{1-\gamma(t)^2}\Delta^{o*} - \frac{2\bar{e}(t)}{1-\gamma(t)^2}^2\right). \tag{A6}$$

Since $\Delta_k^o(t) > \tilde{v}_k^o(t)$, the following inequality can be used.

$$\begin{aligned} &\left(\sum_{k=1}^{h}\frac{4e_{p\,k}(t)\left(\tilde{v}_k^o(t) - \Delta_k^o(t) - e_{p\,k}(t)\right)}{1-\gamma(t)^2} + \frac{2e_p(t)\gamma(t)^2}{1-\gamma(t)^2} + \frac{4\bar{e}(t)}{1-\gamma(t)^2}\Delta^{o*} - \frac{2\bar{e}(t)}{1-\gamma(t)^2}^2\right), \\ &\leq \left(\sum_{k=1}^{h}\frac{-4e_{p_k}^2(t)}{1-\gamma(t)^2} + \frac{2e_p(t)\gamma(t)^2}{1-\gamma(t)^2} + \frac{4\bar{e}(t)}{1-\gamma(t)^2}\Delta^{o*} - \frac{2\bar{e}(t)}{1-\gamma(t)^2}^2\right), \\ &\leq \left(\frac{-4}{1-\gamma(t)^2}e_p(t)^2 + \frac{2e_p(t)\gamma(t)^2}{1-\gamma(t)^2} + \frac{4}{1-\gamma(t)^2}\bar{e}(t)\Delta^{o*} - \frac{2\bar{e}(t)}{1-\gamma(t)^2}^2\right). \end{aligned} \tag{A7}$$

Hence:

$$\begin{aligned} &\lim_{t\longrightarrow\infty}\left(\frac{-4}{1-\gamma(t)^2}e_p(t)^2 + \frac{4}{1-\gamma(t)^2}\bar{e}(t)\Delta^{o*}\right. \\ &\quad \left. -\left(\frac{2}{1-\gamma(t)^2}\right)^2 e_p(t)^2\left(1-\gamma(t)^2\right)\right), \\ &\leq -4\lim_{t\longrightarrow\infty}\left(\frac{1}{1-\gamma(t)^2}e_p(t)\left(e_p(t) - \Delta^{o*}\right)\right) \leq 0. \end{aligned} \tag{A8}$$

Since the above equation is limited and smaller than zero, $\lim_{t\longrightarrow\infty} \sup e_p(t) \leq \Delta^{o*}$. □

B Proof of Weight Convergence for the Hidden Layer

Similar to the proof in Section A, the predictive error of the HCDRNN, $e_p(t)$, is defined as in equation (B.1).

$$\begin{aligned} e_p(t) &= y(t) - \hat{y}(t) + v(t), \\ &= W^{o*}(t)\mathrm{F}^*(S(t)) - W^o(t)F(S(t)) + v^I(t), \\ &= W^{o*}(t)\mathrm{F}^*(S(t)) - W^o(t)\mathrm{F}^* + W^o(t)\mathrm{F}^* \\ &\quad - W^o(t)F(S(t)) + v^I(t). \end{aligned} \tag{B1}$$

F* is the activation function. Equation (B.2) describes the relationship between the predictive error of the neural network, error of the hidden layer $e^I(t)$, and bounded disturbance of the hidden layer $\tilde{v}^I(t)$.

$$\begin{aligned} e_p(t) &= \tilde{v}^I(t) - e^I(t), \\ \tilde{v}^I(t) &= W^{o*}(t)\mathrm{F}^*(S(t)) - W^o(t)\mathrm{F}^* + v^I(t) = \sum_{j=1}^{n} \tilde{v}_j^I(t), \\ e^I(t) &= W^o(t)F(S(t)) - W^o(t)\mathrm{F}^* \\ &= W^o(t)F\left(W^I(t)X(t) + \begin{bmatrix} \left(W_1^D(t) - a.Z_1(t)\right)\Gamma_1(t-1) \\ \vdots \\ \left(W_n^D(t) - a.Z_n(t)\right)\Gamma_n(t-1) \end{bmatrix} \right) \\ &\quad - W^o(t)F\left(W^I(t)X(t) + \begin{bmatrix} \left(W_1^{D*}(t) - a.Z_1(t)\right)\Gamma_1(t-1) \\ \vdots \\ \left(W_n^{D*}(t) - a.Z_n(t)\right)\Gamma_n(t-1) \end{bmatrix} \right). \end{aligned} \tag{B2}$$

Theorem A.2. *Assume that the dead zone vector is defined as* $\Delta^I = [\Delta_1^I \ldots \Delta_j^I \ldots \Delta_h^I]^T$ *in which* $\Delta_j^I(t) \geq F'(t)/\ \varepsilon_{min}^I(t) max\ (|\tilde{v}_j^I(t)|) > 0$ *and* $\varepsilon_{min}^I(t) = min(\varepsilon^I(t)) \neq 0$. *If the weights of the neural network and the dead zone vector is updated such that equation (B.3) is satisfied.*

$$\% \lim_{t\longrightarrow\infty} \sup\left(W_i^I(t+1) - \tilde{W}_i^I(t) - \left(\tilde{\Delta}_i^I(t+1) - \tilde{\Delta}_i^I(t) \right) \right) \leq 0. \tag{B3}$$

Then, the neural network's error is limited as follows and the weight of the hidden layer converges.

$$\lim_{t\longrightarrow\infty} \sup e_p(t) \leq \Delta_i^{I*}, \tag{B4}$$

$\tilde{W}^I(t) = W^I(t) - W^{I*}$, $\tilde{\Delta}_i^I(t) = \Delta_i^I(t) - \Delta_i^{I*}$, Δ_i^{I*}, and W^{I*} are the optimal or suboptimal value of the dead zone vector and weight of the hidden layer.

Proof. By substituting equation (12) and equation (13) in equation (B.3), we have:

$$\begin{aligned} &= \lim_{t\longrightarrow\infty} \left(\frac{4F'_{\min}(t)e_p(t)W^o(t)\tilde{W}^I(t)F'(t)X(t)}{1 - W^o(t)F'(t)X(t)^2} \right) \\ &+ \frac{2F'_{\min}(t)e_p(t)W^o(t)F'(t)X(t)^2}{1 - W^o(t)F'(t)X(t)^2} \\ &- \frac{4F'_{\min}(t)\bar{e}(t)\tilde{\Delta}^I(t)}{1 - W^o(t)F'(t)X(t)^2} - \frac{2F'_{\min}(t)\bar{e}(t)}{1 - W^o(t)F'(t)X(t)^2}^2. \end{aligned} \tag{B5}$$

Since $W^I(t)$ is the inner part of the activation function F, an approximation of F regarding $W^I(t)$ is considered as the product of F and $W^I(t)$. Thus,

$$\begin{aligned} e^I(t) &= W^o(t)\varepsilon^I(t)\tilde{W}^I(t)X(t) \\ &= \sum_j^n \varepsilon_j^I(t)W_j^o(t)\tilde{W}_j^I(t)X_j(t), \\ &= \sum_j^n e_j^I, \end{aligned} \tag{B6}$$

where $\varepsilon^I(t) = [\varepsilon_1^I(t), \varepsilon_2^I(t), \ldots, \varepsilon_j^I(t), \ldots, \varepsilon_n^I(t)]$ and $0 \leq \varepsilon_j^I \leq 1$.

Hence,

$$= \lim_{t\longrightarrow\infty}\left(\frac{4\dot{F}_{\min}(t)e_p(t)F(t)\varepsilon_j^{D^{-1}}(t)}{1-W^o(t)F'(t)X(t)^2}\right.$$

$$+\frac{2\dot{F}_{\min}(t)e_p(t)W_i^o(t)F'(t)X(t)^2}{1-W_i^o(t)F'(t)X(t)^2}$$

$$\left.-\frac{4\dot{F}_{\min}(t)\bar{e}(t)\tilde{\Delta}^I(t)}{1-W_i^o(t)F'(t)X(t)^2}-\frac{2\dot{F}_{\min}(t)\bar{e}(t)}{1-W_i^o(t)F'(t)X(t)^2}^2\right). \tag{B7}$$

By substituting $e_j^I(t)=\tilde{v}_j^I(t)-e_p(t)/n$ and $\Delta^I(t)=\Delta^I(t)-\Delta^{I*}$ in equation (B.7), and considering $\Delta_j^I(t)\geq F'(t)/\varepsilon_{\min}^I(t)\max(|\tilde{v}_j^I(t)|)>0$ and $\varepsilon_{min}^I(t)=\min(\varepsilon^I(t))\neq 0$, we have,

$$\sum_{j=1}^{n}\frac{4\dot{F}_{\min}(t)e_p(t)\left(\tilde{v}_j^I(t)-\left(e_p(t)/n\right)\right)F'(t)\varepsilon^{I^{-1}}(t)}{1-W^o(t)F'(t)X(t)^2}$$

$$+\frac{2\dot{F}_{\min}(t)e_p(t)W^o(t)F'(t)X(t)^2}{1-W^o(t)F'(t)X(t)^2},$$

$$-\frac{4\dot{F}_{\min}(t)\bar{e}(t)\left(\Delta^I(t)-\Delta^{I*}\right)}{1-W^o(t)F'(t)X(t)^2}-\frac{2\dot{F}_{\min}(t)\bar{e}(t)}{1-W^o(t)F'(t)X(t)^2}^2,$$

$$\leq\sum_{j=1}^{n}\frac{-4\dot{F}_{\min}(t)e_p(t)}{1-W^o(t)F'(t)X(t)^2}\left[\frac{F'(t)e_p(t)}{n\varepsilon^I(t)}-\frac{F'(t)\tilde{v}_j^I(t)}{\varepsilon^I(t)}+\Delta_j^I(t)\right],$$

$$+\frac{2\dot{F}_{\min}(t)e_p(t)W^o(t)F'(t)X(t)^2}{1-W^o(t)F'(t)X(t)^2}+\frac{4\dot{F}_{\min}(t)\bar{e}(t)\left(\Delta^{I*}\right)}{1-W^o(t)F'(t)X(t)^2}$$

$$-\frac{2\dot{F}_{\min}(t)\bar{e}(t)}{1-W^o(t)F'(t)X(t)^2}^2,$$

$$\leq\frac{-4\dot{F}_{\min}(t)}{1-W^o(t)F'(t)X(t)^2}e_p(t)^2$$

$$+\frac{4\dot{F}_{\min}(t)}{1-W^o(t)F'(t)X(t)^2}\bar{e}(t)\left(\Delta^{I*}\right)^2,$$

$$-\frac{-4\dot{F}_{\min}(t)}{\left(1-W^o(t)F'(t)X(t)^2\right)^2}e_p(t)^2\left(1-W^o(t)F'(t)X(t)^2\right),$$

$$=\frac{-4\dot{F}_{\min}(t)}{1-W^o(t)F'(t)X(t)^2}e_p(t)\left(e_p(t)-\Delta^{I*}+e_p(t)\right). \tag{B8}$$

Hence,

$$\lim_{t\longrightarrow\infty}\left(\frac{-4\dot{F}_{\min}(t)}{1-W^o(t)F'(t)X(t)^2}e_p(t)\left(e_p(t)-\Delta^{I*}+e_p(t)\right)\right), \tag{B9}$$

$$\leq -4\lim_{t\longrightarrow\infty}\left(\frac{\dot{F}_{\min}(t)}{1-W^o(t)F'(t)X(t)^2}e_p(t)\left(e_p(t)-\Delta^{I*}\right)\right)\leq 0.$$

Since the above equation is limited and smaller than zero, $\lim_{t\longrightarrow\infty}\sup e_p(t)\leq\Delta_i^{I*}$. □

C Proof of Weight Convergence for the Contex Layer

Similar to the proof in Sections A and B, the predictive error of the HCDRNN, $e_p(t)$, is defined as in the following equation :

$$\begin{aligned}e_p(t)&=y(t)-\hat{y}(t)+v(t)=W^{o*}(t)\mathrm{F}^*(S(t))\\&\quad-W^o(t)F(S(t))+v^D(t),\\&=W^{o*}(t)\mathrm{F}^*(S(t))-W^o(t)\mathrm{F}^*+W^o(t)\mathrm{F}^*\\&\quad-W^o(t)F(S(t))+v^D(t).\end{aligned} \tag{C1}$$

Equation (C.2) describes the relationship between the predictive error of the neural network, error of the contex layer $e^D(t)$, and bounded disturbance of the contex layer $\tilde{v}^D(t)$.

$$e_p(t) = \tilde{v}^D(t) - e^D(t), \tilde{v}^D(t) = W^{o*}(t)\mathrm{F}^*(S(t)) - W^o(t)\mathrm{F}^* + v^D(t) = \sum_{j=1}^{n} \tilde{v}_j^D(t),$$

$$e^D(t) = W^o(t)F(S(t)) - W^o(t)\mathrm{F}^* = W^o(t)F\left(W^I(t)X(t) + \begin{bmatrix} \left(W_1^D(t) - a.Z_1(t)\right)\Gamma_1(t-1) \\ \vdots \\ \left(W_n^D(t) - a.Z_n(t)\right)\Gamma_n(t-1) \end{bmatrix}\right), \tag{C2}$$

$$- W^o(t)F\left(W^I(t)X(t) + \begin{bmatrix} \left(W_1^{D*}(t) - a.Z_1(t)\right)\Gamma_1(t-1) \\ \vdots \\ \left(W_n^{D*}(t) - a.Z_n(t)\right)\Gamma_n(t-1) \end{bmatrix}\right).$$

Theorem A.3. *Assume that the dead zone vector is defined as* $\Delta^D = [\Delta_1^D \ldots \Delta_j^D \ldots \Delta_h^D]^T$ *in which* $\Delta_j^D(t) \geq F'(t)/\varepsilon_{min}^D(t)max(|\tilde{v}_j^D(t)|) > 0$ *and* $\varepsilon_{min}^D(t) = min(\varepsilon^D(t)) \neq 0$. *If the weights of the neural network and the dead zone vector is updated such that equation (C.3) is satisfied.*

$$\lim_{t\longrightarrow\infty} \sup\left(\tilde{W}_i^D(t+1) - \tilde{W}_i^D(t) - \left(\tilde{\Delta}_i^D(t+1) - \tilde{\Delta}_i^D(t)\right)\right) \leq 0. \tag{C3}$$

Then, the neural network's error is limited as follows and the weight of the contex layer converges.

$$\lim_{t\longrightarrow\infty} \sup e_p(t) \leq \Delta_i^{D*}, \tag{C4}$$

$\tilde{W}_i^D(t) = W_i^D(t) - W_i^{D*}$, $\tilde{\Delta}_i^D(t) = \Delta_i^D(t) - \Delta_i^{D*}$, Δ^{D*}, *and* W_i^{D*} *are the optimal or suboptimal value of the dead zone vector of the contex layer.*

Proof. By substituting equation (14) and equation (15) in equation (C.3), we have,

$$\lim_{t\longrightarrow\infty}\left(\frac{4F'_{\min}(t)e_p(t)e_j^D\varepsilon_j^{D^{-1}}(t)\left(1-\gamma_i^2(t)\right)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2} + \frac{2F'_{\min}(t)e_p(t)W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2} - \frac{4F'_{\min}(t)\tilde{\Delta}_i^D(t)\bar{\mathrm{e}}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}\right.$$

$$\left. - \frac{2F'_{\min}(t)\bar{\mathrm{e}}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}^2\right). \tag{C5}$$

Since $W^D(t)$ is the inner part of the activation function F, an approximation of F regarding $W^D(t)$ is considered as the product of F and $W^D(t)$. Thus,

$$\begin{aligned} e^D(t) &= W^o(t)\varepsilon^D(t)\tilde{W}^D(t)\Gamma(t-1) = \sum_j^n \varepsilon_j^D(t)W_j^o(t)\tilde{W}_j^D(t)\Gamma_j(t-1), \\ &= \sum_j^n e_j^D, \end{aligned} \tag{C6}$$

where $\varepsilon^D(t) = [\varepsilon_1^D(t), \varepsilon_2^D(t), \ldots, \varepsilon_j^D(t), \ldots, \varepsilon_n^D(t)]$, $0 \le j \le n$, and $0 \le \varepsilon_j^D \le 1$.

$$\lim_{t\longrightarrow\infty}\left(\frac{4F_{\min}^{'}(t)e_p(t)e_j^D\varepsilon_j^{D^{-1}}(t)\left(1-\gamma_i^2(t)\right)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}+\frac{2F_{\min}^{'}(t)e_p(t)W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}-\frac{4F_{\min}^{'}(t)\widetilde{\Delta}_i^D(t)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}\right.$$
$$\left.-\frac{2F_{\min}^{'}(t)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}^2\right). \tag{C7}$$

By substituting $e_j^D(t) = \widetilde{v}_j^D(t) - e_p(t)/n$, $\widetilde{\Delta}_i^D(t) = \Delta_i^D(t) - \Delta_i^{D*}$ in equation (C.7) and considering $\Delta_i^D(t) \ge F'(t)/\varepsilon_{\min}^D(t)\max(|\widetilde{v}_j^D(t)|) > 0$ and $\varepsilon_{\min}^D(t) = \min(\varepsilon^D(t)) \ne 0$, the following inequality can be used.

$$\begin{aligned}
&\sum_{j=1}^{n}\frac{4F_{\min}^{'}(t)e_p(t)\left(\widetilde{v}_j^D(t)-e_p(t)/n\right)\varepsilon_j^{D^{-1}}(t)\left(1-\gamma_i^2(t)\right)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}+\frac{2F_{\min}^{'}(t)e_p(t)W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2},\\
&-\frac{4F_{\min}^{'}(t)\left(\Delta_i^D(t)-\Delta_i^{D*}\right)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}-\frac{2F_{\min}^{'}(t)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}^2,\\
&\le\sum_{j=1}^{n}\frac{-4F_{\min}^{'}(t)e_p(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}\left[\left(1-\gamma_i^2(t)\right)\frac{e_p(t)}{n\varepsilon_j^D}-\frac{\left(1-\gamma_i^2(t)\right)\widetilde{v}_j^D(t)}{\varepsilon_j^D}+\Delta_i^D(t)\right],\\
&+\frac{2F_{\min}^{'}(t)e_p(t)W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}+\frac{4F_{\min}^{'}(t)\left(\Delta_i^{D*}\right)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2},\\
&-\frac{2F_{\min}^{'}(t)\overline{e}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}^2\le\frac{-4F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}e_p(t)^2,\\
&+\frac{4F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}\overline{e}(t)\left(\Delta_i^{D*}\right)^2,\\
&-\frac{4F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}e_p(t)^2\left(1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2\right),\\
&=\frac{-4F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}e_p(t)\left(e_p(t)-\Delta_i^{D*}+e_p(t)\right).
\end{aligned} \tag{C8}$$

Hence,

$$\begin{aligned}
&\lim_{t\longrightarrow\infty}\left(\frac{-4F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}e_p(t)\left(e_p(t)-\Delta_i^{D*}+e_p(t)\right)\right),\\
&\le-4\lim_{t\longrightarrow\infty}\left(\frac{F_{\min}^{'}(t)}{1-W_i^o(t)\left(1-\gamma_i^2(t)\right)\Gamma_i^T(t-1)^2}e_p(t)\left(e_p(t)-\Delta_i^{D*}\right)\right)\le 0.
\end{aligned} \tag{C9}$$

Since the above equation is limited and smaller than zero, $\lim_{t\longrightarrow\infty} \sup e_p(t) \leq \Delta^{D*}$. □

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest in preparing this article.

References

[1] B. Inner and S. Kucuk, "A novel kinematic design, analysis and simulation tool for general Stewart platforms," *SIMULATION*, vol. 89, no. 7, pp. 876–897, 2013.

[2] X. Huang and B. Cui, "A neural dynamic system for solving convex nonlinear optimization problems with hybrid constraints," *Neural Computing and Applications*, vol. 31, pp. 1–12, 2018.

[3] R. Kumar, S. Srivastava, and J. Gupta, "Diagonal recurrent neural network based adaptive control of nonlinear dynamical systems using lyapunov stability criterion," *ISA Transactions*, vol. 67, pp. 407–427, 2017.

[4] J. Qiao, X. Meng, and W. Li, "An incremental neuronal-activity-based RBF neural network for nonlinear system modeling," *Neurocomputing*, vol. 302, pp. 1–11, 2018.

[5] Z. Chen, F. Huang, W. Sun, J. Gu, and B. Yao, "RBF-neural-network-based adaptive robust control for nonlinear bilateral teleoperation manipulators with uncertainty and time delay," *IEEE*, vol. 25, no. 2, pp. 906–918, 2020.

[6] S. M. Lu, D. P. Li, and Y. J. Liu, "Adaptive neural network control for uncertain time-varying state constrained robotics systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 12, pp. 2511–2518, 2019.

[7] Y. Chu, J. Fei, and S. Hou, "Adaptive global sliding-mode control for dynamic systems using double hidden layer recurrent neural network structure," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 4, pp. 1297–1309, 2020.

[8] Z. Chen, F. Huang, W. Chen et al., "RBFNN-based adaptive sliding mode control design for delayed nonlinear multilateral telerobotic system with cooperative manipulation," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 2, pp. 1236–1247, 2020.

[9] C.-S. Chen, "Robust self-organizing neural-fuzzy control with uncertainty observer for MIMO nonlinear systems," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 4, pp. 694–706, 2011.

[10] H. Han and J. Qiao, "Nonlinear model-predictive control for industrial processes: an application to wastewater treatment process," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 4, pp. 1970–1982, 2014.

[11] H.-G. Han, L. Zhang, Y. Hou, and J.-F. Qiao, "Nonlinear model predictive control based on a self-organizing recurrent neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 2, pp. 402–415, 2016.

[12] K. Aihara, T. Takabe, and M. Toyoda, "Chaotic neural networks," *Physics Letters A*, vol. 144, no. 6-7, pp. 333–340, 1990.

[13] J. Li, F. Liu, Z.-H. Guan, and T. Li, "A new chaotic Hopfield neural network and its synthesis via parameter switchings," *Neurocomputing*, vol. 117, pp. 33–39, 2013.

[14] R. Mazrooei-Sebdani and S. Farjami, "RETRACTED: Bifurcations and Chaos in a Discrete-Time-Delayed Hopfield Neural Network with Ring Structures and Different Internal Decays," *Neurocomputing*, vol. 99, pp. 154–162, 2013.

[15] L. Wang, Z. Meng, Y. Sun, L. Guo, and M. Zhou, "Design and analysis of a novel chaotic diagonal recurrent neural network," *Communications in Nonlinear Science and Numerical Simulation*, vol. 26, no. 1-3, pp. 11–23, 2015.

[16] A. Taherkhani, S. A. Seyyedsalehi, and A. Mohammadi, "Design of chaotic neural network for robust phoneme recognition," in *Proceedings of the 2008 IEEE International Symposium on Signal Processing and Information Technology*, pp. 106–110, IEEE, Sarajevo, Bosnia and Herzegovina, December, 2008.

[17] A. Taherkhani, S. A. Seyyedsalehi, and A. Jafari, "Design of a chaotic neural network for training and retrieval of grayscale and binary patterns," *Neurocomputing*, vol. 74, no. 17, pp. 2824–2833, 2011.

[18] W. Dongsu and G. Hongbin, "Adaptive sliding control of six-DOF flight simulator motion platform," *Chinese Journal of Aeronautics*, vol. 20, no. 5, pp. 425–433, 2007.

[19] M. Ghorbani and S. K. H. Sani, "Fuzzy nonlinear predictive control of Stewart platform," in *Proceedings of the 2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, pp. 383–389, IEEE, Qazvin, Iran, January, 2016.

[20] L. Jin, S. Li, J. Yu, and J. He, "Robot manipulator control using neural networks: a survey," *Neurocomputing*, vol. 285, pp. 23–34, 2018.

[21] B. Kouvaritakis and M. Cannon, *Model predictive control*, Vol. 38, Springer International Publishing, Switzerland, 2016.

[22] L. Grüne and J. Pannek, "Nonlinear model predictive control," in *Nonlinear Model Predictive Control*, pp. 45–69, Springer, Cham, 2017.

[23] N. Nikdel, P. Nikdel, M. A. Badamchizadeh, and I. Hassanzadeh, "Using neural network model predictive control for controlling shape memory alloy-based manipulator," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 3, pp. 1394–1401, 2014.

[24] H. Peng, J. Wu, G. Inoussa, Q. Deng, and K. Nakano, "Nonlinear system modeling and predictive control using the RBF nets-based quasi-linear ARX model," *Control Engineering Practice*, vol. 17, no. 1, pp. 59–66, 2009.

[25] Y. Pan and J. Wang, "Model predictive control of unknown nonlinear dynamical systems based on recurrent neural networks," *IEEE Transactions on Industrial Electronics*, vol. 59, no. 8, pp. 3089–3101, 2012.

[26] R. Al Seyab and Y. Cao, "Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation," *Journal of Process Control*, vol. 18, no. 6, pp. 568–581, 2008.

[27] L.-W. Tsai, "Solving the inverse dynamics of a Stewart-Gough manipulator by the principle of virtual work," *Journal of Mechanical Design*, vol. 122, no. 1, pp. 3–9, 2000.

[28] A. Afram, F. Janabi-Sharifi, A. S. Fung, and K. Raahemifar, "Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: a state of the art review and case study of a residential HVAC system," *Energy and Buildings*, vol. 141, pp. 96–113, 2017.

[29] S. Johari, M. Yaghoobi, and H. R. Kobravi, "Nonlinear model predictive control based on hyper chaotic diagonal recurrent neural network," *Journal of Central South University*, vol. 29, no. 1, pp. 197–208, 2022.

[30] C.-C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 144–156, 1995.

[31] H.-G. Han, X.-L. Wu, and J.-F. Qiao, "Real-time model predictive control using a self-organizing neural network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 24, no. 9, pp. 1425–1436, 2013.

[32] J. Wang and C. M. Gosselin, "A new approach for the dynamic analysis of parallel manipulators," *Multibody System Dynamics*, vol. 2, no. 3, pp. 317–334, 1998.

[33] M. S. Ayas, E. Sahin, and I. H. Altas, "High order differential feedback controller design and implementation for a Stewart platform," *Journal of Vibration and Control*, vol. 26, no. 11-12, pp. 976–988, 2020.

[34] G. Lebret, K. Liu, and F. L. Lewis, "Dynamic analysis and control of a Stewart platform manipulator," *Journal of Robotic Systems*, vol. 10, no. 5, pp. 629–655, 1993.