Journal of Control Science and Engineering

Advances in Methods for Networked and Cyber-Physical System

Guest Editors: Jason Gu, Xiaomei Qi, Ying Wang, Fei Liu, and Chengjin Zhang

Advances in Methods for Networked and Cyber-Physical System

Journal of Control Science and Engineering

Advances in Methods for Networked and Cyber-Physical System

Guest Editors: Jason Gu, Xiaomei Qi, Ying Wang, Fei Liu, and Chengjin Zhang

Editorial Board

Contents

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2014, Article ID 496075, 2 pages
http://dx.doi.org/10.1155/2014/496075

Editorial

Advances in Methods for Networked and Cyber-Physical System

Jason Gu,[1] Xiaomei Qi,[2] Ying Wang,[3] Fei Liu,[4] and Chengjin Zhang[5]

[1] *Department of Electrical and Computer Engineering, Dalhousie University, Halifax, NS, Canada B3J 2X4*
[2] *College of Electrical and Electronic Engineering, Shandong University of Technology, Zibo 255049, China*
[3] *Department of Electrical and Mechatronics Engineering, Southern Polytechnic State University, 1100 South Marietta Parkway Marietta, Marietta, GA 30060, USA*
[4] *Computer Science and Computer Engineering, La Trobe University, Melbourne, VIC 3086, Australia*
[5] *School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai West Road 180, Weihai 264209, China*

Correspondence should be addressed to Jason Gu; jason.gu@dal.ca

Received 14 April 2014; Accepted 14 April 2014; Published 29 May 2014

Networked and cyber-physical system (NCPS) represents a system which tightly integrates computation, communication, and physical processes. Typical NCPS is open, dynamic, and heterogeneous in many dimensions and often needs to be rapidly instantiated and deployed for a given mission. NCPS can be found in areas as diverse as aerospace, automotive, chemical processes, flexible manufacturing systems, medical systems, unmanned vehicles, and traffic control. Despite its high complexity, NCPS has many challenges both in theory and applications. The main focus of this special issue is on the most recent theoretical developments, results, and applications in the area of methods for NCPS.

This special issue collected seven papers that represent a sample of current developments in networked and cyber-physical system. The paper entitled "*JAUS to EtherCAT bridge: toward real-time and deterministic joint architecture for unmanned systems*" presents a basic joint architecture for unmanned systems (JAUS) implementation in order to evaluate the performance of the standard at its core. By using EtherCAT for internal controls and JAUS as the subsystem level protocol, both interoperability and performance can be achieved and demonstrated in the paper.

The paper with title "*Improved weighted Shapley value model for the fourth party logistics supply chain coalition*" first analyzes the fourth party logistics supply chain coalition profit allocation models, the classical Shapley value method. Then the weight of individual enterprise in the coalition by the analytic hierarchy process is analyzed. The numerical study shows that the profit allocation method combining Shapley value and distribution according to contribution is relatively rational and practical and is a useful profit allocation mechanism for the fourth party logistics supply chain coalition.

The paper titled "*Mobile robot path planning using polyclonal-based artificial immune network*" presents polyclonal-based artificial immune network (PC-AIN) for mobile robot path planning. Immunity polyclonal algorithm (IPCA) increases the diversity of antibodies which tend to the same extreme value and finally selects the antibody with the highest concentration. Meanwhile, immunity polyclonal algorithm effectively solves the problem of local minima caused by artificial potential field during the structure of parameter in artificial immune network. Extensive experiments show that the proposed method not only solves immature convergence problem of artificial immune network but also overcomes local minima problem of artificial potential field. So, mobile robot can avoid obstacles, escape traps, and reach the goal with optimum path and faster convergence speed.

The paper titled "*Integrity design for networked control systems with actuator failures and data packet dropouts*" presents the integrity design problem of fault tolerant control for networked control system (NCS) with actuator failures and data packet dropouts. The data packet dropouts in both sensor-controller (S-C) and controller-actuator (C-A) links are described by two switches, which can be modeled as

a discrete event system with known rate. After introducing the matrix of actuator failure, the closed-loop NCS is developed, which can be viewed as asynchronous dynamical systems (ADSs). Then, the sufficiency of exponential stability for the NCS is obtained based on the theory of ADSs. The output feedback controllers that can guarantee system stability are also proposed. Finally, two numerical examples are given to demonstrate the validity of our proposed approach.

The paper with title "*Research on feature extraction of indicator card data for sucker-rod pump working condition diagnosis*" studies three feature extraction methods of sucker-rod pump indicator card data, which are Fourier Descriptors (FD), Geometric Moment Vector (GMV), and Gray Level Matrix Statistics (GLMX), respectively. Numerical experiments show that the Fourier Descriptors algorithm requires less running time and less memory space with possible loss of information due to nonoptimal numbers of Fourier Descriptors, the Geometric Moment Vector algorithm is more time consuming and requires more memory space, while the Gray Level Matrix Statistics algorithm provides low-dimensional feature vectors with more time consumed and more memory space. Furthermore, the characteristic of rotational invariance, both in the Fourier Descriptors algorithm and the Geometric Moment Vector algorithm, may result in improper pattern recognition of indicator card data when used for sucker-rod pump working condition diagnosis.

The paper "*Real-time detection of application-layer DDoS attack using time series analysis*" presents a novel approach to detect application-layer DDoS attack based on entropy of HTTP GET requests per source IP address (HRPI). By approximating the adaptive autoregressive (AAR) model, the HRPI time series is transformed into a multidimensional vector series. Then a trained support vector machine (SVM) classifier is applied to identify the attacks. The experiments with several databases are performed and results show that this approach can detect application-layer DDoS attacks effectively.

The paper "*An improved differential evolution algorithm based on adaptive parameter*" proposes an adaptive parameter adjustment method which can dynamically adjust control parameters according to the evolution stage. The experiments on high dimensional function optimization showed that the improved algorithm has more powerful global exploration ability and faster convergence speed.

Journal of Control Science and Engineering will continue to attract and disseminate original, theoretically advanced, and practically advantaged papers pertinent to Control Science and Engineering.

Jason Gu
Xiaomei Qi
Ying Wang
Fei Liu
Chengjin Zhang

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2014, Article ID 631487, 20 pages
http://dx.doi.org/10.1155/2014/631487

Research Article

JAUS to EtherCAT Bridge: Toward Real-Time and Deterministic Joint Architecture for Unmanned Systems

Jie Sheng,[1] Sam Chung,[1] Leo Hansel,[1] Don McLane,[1] Joel Morrah,[1] Seung-Ho Baeg,[2] and Sangdeok Park[2]

[1] *Institute of Technology, University of Washington, Tacoma, WA, USA*
[2] *Division of Applied Robot Technology, Korea Institute of Industrial Technology, Republic of Korea*

Correspondence should be addressed to Jie Sheng; shengj2@uw.edu

Received 26 June 2013; Revised 12 December 2013; Accepted 26 December 2013; Published 27 April 2014

Academic Editor: Jason Gu

The Joint Architecture for Unmanned Systems (JAUS) is a communication standard that allows for interoperability between Unmanned Vehicles (UVs). Current research indicates that JAUS-compliant systems do not meet real-time performance guidelines necessary for internal systems in UVs. However, there is a lack of quantitative data illustrating the performance shortcomings of JAUS or clear explanations on what causes these performance issues or comparisons with existing internal communication systems. In this research, we first develop a basic C++ implementation of JAUS and evaluate its performance with quantitative data and compare the results with published performance data of Controller Area Network (CAN) to determine the feasibility of the JAUS standard. Our results indicate that the main reason of JAUS's poor performance lies in the latency inherent in the hierarchical structure of JAUS and the overhead of User Datagram Protocol (UDP) messages, which has been used with JAUS and is slower than the high-speed CAN. Additionally, UDP has no scheduling mechanism, which makes it virtually impossible to guarantee messages meeting their deadlines. Considering the slow and nondeterministic JAUS communication from subsystems to components, which is JAUS Level 3 compliance, we then propose a solution by bringing Ethernet for Control Automation Technology (EtherCAT) to add speed, deterministic feature, and security. The JAUS-EtherCAT mapping, which we called a JEBridge, is implemented into nodes and components. Both quantitative and qualitative results are provided to show that JEBridge and JAUS Level 3 compliance can bring not only interoperability but also reasonable performance to UVs.

1. Introduction

The United States Congress has mandated that by 2015 30% of all military vehicles must be Unmanned Vehicles (UVs) [1]. One major roadblock, however, is that most UVs are made by various manufacturers and come in many different makes and models. Communication with UVs, either between the UV and an Operator Control Unit (OCU) or another UV, is typically proprietary. As a result, UVs that coordinate on missions must be specifically designed to communicate with each other. If a specific UV necessary for a mission is unavailable, there is no way to substitute another UV. In addition, OCUs come in many forms and the person charged with operating these vehicles must learn a new interface for each UV.

The Joint Architecture for Unmanned Systems (JAUS) is an initiative by the United States Department of Defense (DoD) to deal with interoperability issues between UVs. If all UVs used the same communication standard, any UV capable of performing a mission could be used, regardless of the manufacturer. By standardizing communication in UVs with the same functionality and commands, OCUs could be more consistently designed or even customized for the preference of operators.

The JAUS Working Group (JWG) was the first body tasked with developing and maintaining the JAUS standard [2]. Since then, the Society for Automotive Engineers (SAE) has taken over responsibility for JAUS [3].

Interoperability between UVs is an enticing prospect but there is another aspect to consider, performance. JAUS, first and foremost, is for use in mission critical vehicles for the military. Communication is not limited to tactical information such as mission planning. JAUS also has been

designed for facilitating communication between the control systems of UVs such as motors and braking systems, among others. Although interoperability is enticing, it cannot trade off with performance. Previous research indicates that real-time communication necessary in UVs cannot be achieved with JAUS in its current state [2, 4]. Other work regarding JAUS focuses on the interoperability aspect and does not mention issues with performance [5, 6]. In either case, quantitative data illustrating the performance shortcomings of JAUS are not present. But even with hard data we need performance measurements that are necessary in real-time, mission-critical vehicles with which we compare our results. JAUS is meant to be used in a variety of ground, air, and sea UVs, including passenger-style vehicles. As such, in order for JAUS to be considered as a communication standard for internal controls inside this type of vehicle, it would need to perform the same as, or better than, the current communication system which is used inside passenger vehicles.

This paper will investigate two main problems.

(1) Is JAUS a suitable communication standard for internal controls of passenger vehicles? If not, can quantitative data be presented to show the performance shortcomings of JAUS?

(2) If the JAUS is used to guarantee the interoperability between UVs, is there a solution to improve the communication performance?

Correspondingly, the two main contributions of the paper are as follows.

(1) We provide quantitative data illustrating that JAUS lacks real-time facilities, which was claimed (without the data support though) in [7]. The Controller Area Network (CAN) is the de facto standard for internal communication in passenger vehicles, controlling everything from luxuries like automatic windows and locks to necessities such as engine speed and antilock brakes [8]. By comparing the performance of JAUS based upon User Datagram Protocol (UDP) with that of CAN, the shortcomings of JAUS are quantified, and a seminal work [9] in the field of performance analysis of CAN systems is used as a benchmark for the performance analysis of JAUS.

(2) We design a JAUS-EtherCAT mapping, which we called a JEBridge, and implement it as well as its communication into JAUS Compliance Level 3 to improve the performance of JAUS communication. Due to the fact that JAUS lacks real-time facilities, but real-time, closed loop control design requires deterministic timing constraints, we utilize Ethernet for Control Automation Technology (EtherCAT) to avoid message traffic and thus improve the transfer rate. We provide scientific reference both quantitative and qualitative that shows how EtherCAT can help mitigate some issues on JAUS.

We note here that JAUS is a higher level protocol than CAN; so one could implement a JAUS layer over a CAN link/physical layer. However, the CAN bus is slow and has no real-time guarantees, and mapping the rich variety of JAUS messages to CAN's limited data fields is awkward. EtherCAT, however, can offer real-time guarantees and is faster. That is the motivation behind our JEBridge solution.

The rest of this paper proceeds as follows. Section 2 examines related work in the areas of JAUS and CAN and JAUS performance improvement. A brief review of concepts on nondeterministic and deterministic control, as well as EtherCAT, will be given in Section 3, followed by a detailed description about how CAN works in Section 4. Section 4 also discusses the two CAN research papers and their results, which will be used for comparison with the results of our experiments with JAUS. A detailed introduction of JAUS is given in Section 5. Section 6 explores the JAUS standard in greater detail first, then discusses our JAUS implementation, and further outlines the tests we perform for our experiments and the results we obtained. In Section 7, we first introduce JAUS Level 3 Compliance using JEBridge and its implementation; we then analyze the performance and the deterministic control of the JAUS Level 3 Compliance architecture using JEBridge. Section 8 concludes this paper.

2. Related Work

Current JAUS research indicates that performance is a significant problem. In [10], the authors created the KUL-1 gas-powered vehicle to test their JAUS implementation. The vehicle has navigation control, obstacle detection, path planning, and vehicle control capabilities. The authors point out that the latency in the vehicle is unacceptably high.

Efforts have been made to improve the JAUS performance. The authors of [4] created a partially JAUS-compliant communication framework for UVs. The primary focus of the framework is communication internal to a UV as opposed to communication between UVs. The authors chose not to implement a fully JAUS-compliant framework due to latency issues with JAUS. The issue of latency in fully JAUS-compliant systems also is noted in [2]. The authors created a UV called the Armadillo. Due to high latency caused by routing messages through the node, the authors removed it, opting to route messages directly to components. This conflicts with interoperability and full JAUS compliance but gave Armadillo improved performance. In [10], the authors point out that the latency in the vehicle is unacceptably high.

Clearly, creating fully JAUS-compliant vehicles is a difficult task. The primary purpose for JAUS, at least at this stage of development, is interoperability and not performance, since the JAUS documentation makes no mention of performance requirements [2]. In order for JAUS to be adopted as a viable communication standard, performance must be a priority. To assess the performance of JAUS-compliant systems, we must first obtain quantitative data in regards to the performance of JAUS. Then we need a suitable performance benchmark representative of performance requirements in vehicles today.

The benchmark we will use is CAN. The work in [9] is important in the field of CAN performance. In this paper, Tindell and Burns adapt CPU scheduling techniques [11] to

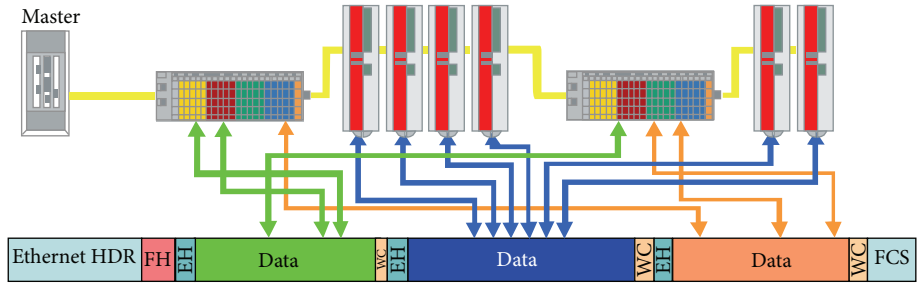


Figure 1: EtherCAT frame processing [13].

the problem of bounding performance in CAN systems. The results of their work will be the performance standard that we will use to assess the feasibility of JAUS as a communication standard for UVs.

We note that Albus et al. [12] developed a reference model for Unmanned Vehicle systems called 4D/RCS. They stated that 4D/RCS architecture is naturally adaptable to the DoD/Army standards in a combined domain of vehicle systems, combat support, and software engineering. 4D/RCS provides a methodology by which military systems, that meet the operational requirements in the Joint Architecture for Unmanned Ground Systems (JAUGS) Domain Models, can be engineered to meet the performance specifications defined in the JAUGS Reference Architecture. Unlike JAUS, which does not provide the guidelines for functionally organizing components, 4D/RCS describes in detail the functions and associated interfaces necessary for each node to provide sensory processing, world modeling, knowledge management, value judgment, and behavior generation. Furthermore, it describes that the functional loop should be replicated throughout all the nodes inside a system. In fact, 4D/RCS can serve as organizational frameworks for the JAUS components and messages, which, however, is beyond the scope of this research work.

3. Background

In this section, we will briefly review the concepts of deterministic control and EtherCAT. We will discuss JAUS and CAN in more details in separate sections soon.

3.1. Deterministic Control. In a local area network using User Data Protocol (UDP) over Ethernet, timing is nondeterminate. Despite that, on the time scales in which humans perceive events, the internet protocol is generally fast enough. For lower level systems, however, the situation is different. Latency still has to be good enough. But here "good enough" is far more stringent. For instance, motion controls of robotic arms or traction motors have closed loop control cycles in the millisecond range. They continually sample sensors and make adjustments to actuator drive. Deterministic timing with bounded latency is an absolute requirement.

Through our research we found that JAUS's poor performance lies in the latency inherent in the hierarchical structure of JAUS and the overhead of UDP messages, which has been used behind JAUS. Moreover, UDP has no scheduling mechanism, which makes it virtually impossible to guarantee messages meeting their deadlines and thus cannot meet the deterministic timing requirement for lower level closed loop control systems. Considering the slow and nondeterministic JAUS communication from subsystems to components, our solution is to introduce Ethernet for Control Automation Technology (EtherCAT) to add speed, deterministic feature, and security.

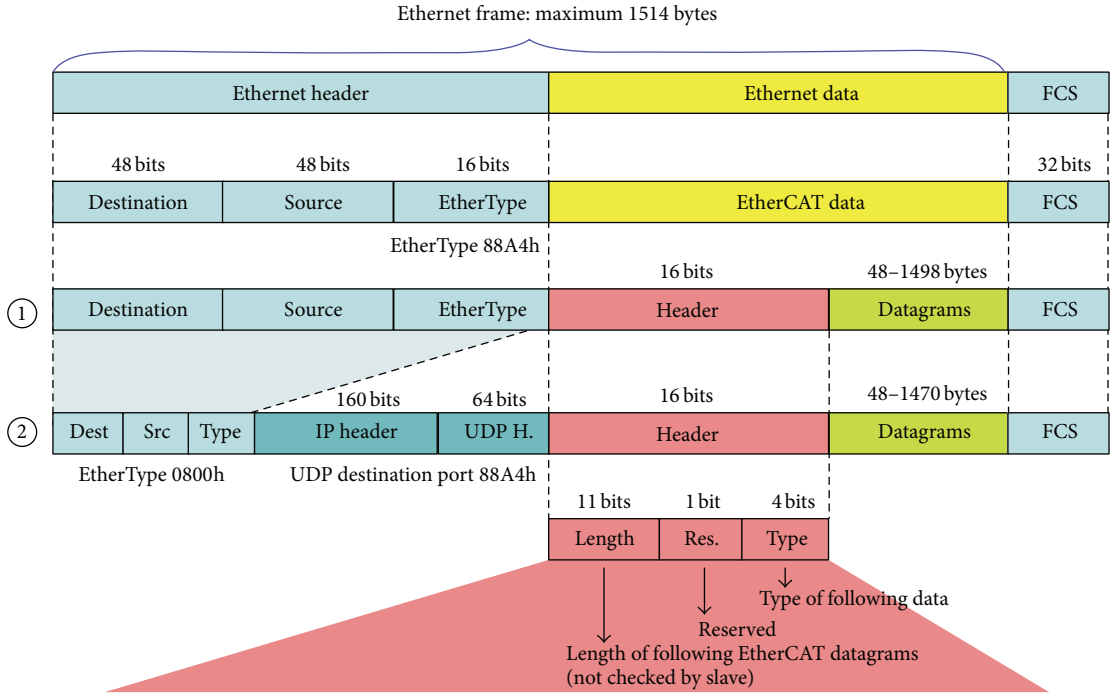
3.2. EtherCAT. EtherCAT is a real-time, high speed, and flexible Ethernet based protocol. In comparison to other Ethernet based communication solutions, EtherCAT utilizes the available full duplex bandwidth in a very efficient way because it implements a "processing on the fly" approach. The Ethernet frames in Figure 1 are sent by a master device and read and written by all EtherCAT slave devices while they are passed from one device to the next [13].

The EtherCAT protocol is optimized for process data which is embedded in the standard IEEE 802.3; see Figure 2 for Ethernet and EtherCAT frame structure.

The Ethernet header consists of Destination (6 bytes), Source (6 bytes), and EtherType (2 bytes). Also, it shows an Ethernet frame using the Ether type 0x88A4. We are interested in the first case (the circled number 1 in Figure 2), which is the EtherCAT protocol. The second case (the circled number 2 in Figure 2) is for a case when we want to send an EtherCAT packet over a UDP. The EtherCAT protocol consists of the EtherCAT protocol header (2 bytes) which contains the EtherCAT frame size in bytes (11 bits) and a protocol type (4 bits set to 1 for EtherCAT) followed by EtherCAT telegrams, which is shown in Figure 3. Each EtherCAT telegram starts with a telegram header (10 bytes) followed by the process dataand is terminated with a working counter (2 bytes) [7, 13].

4. Controller Area Network (CAN)

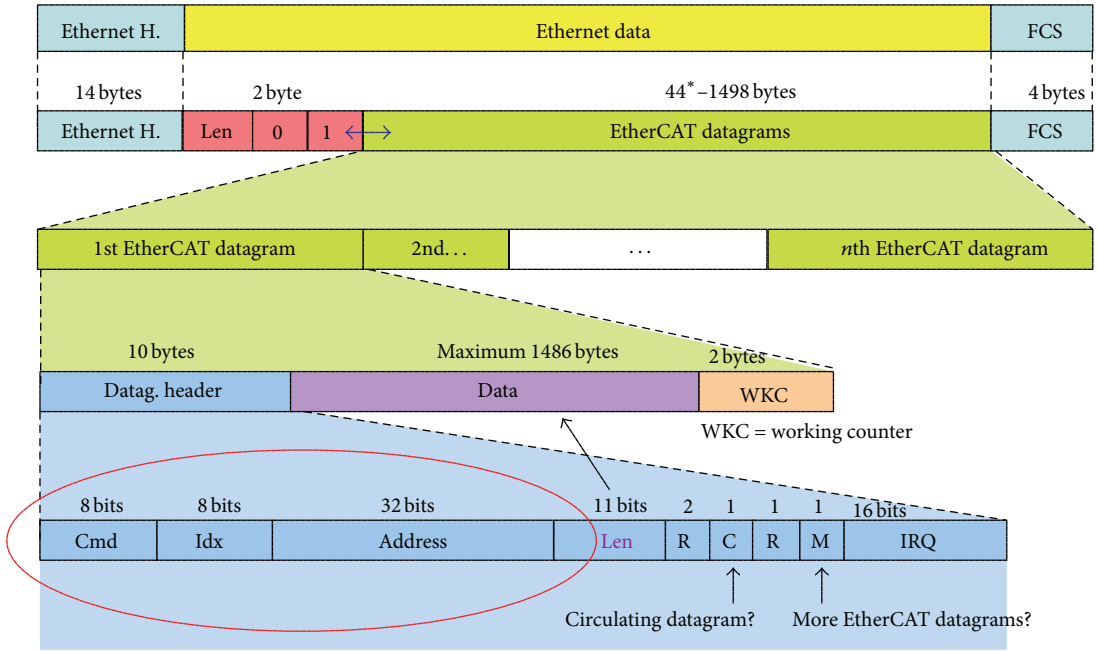
4.1. The CAN Standard. The Bosch company began development of CAN in 1983 [14]. By providing a standardized bus communication network, the problems brought by point-to-point communication using wires have been greatly reduced [8, 15]. As a serial communication standard used in passenger vehicles, CAN supports communication between Electronic Control Units (ECU) interfacing with sensors and actuators



1 Simple EtherCAT communication

2 EtherCAT communication over internet

Figure 2: EtherCAT frame structure (slide 28 of [7]).



*add 1–32 padding bytes if Ethernet frame is less than 64

Figure 3: EtherCAT datagram (slide 33 of [7]).

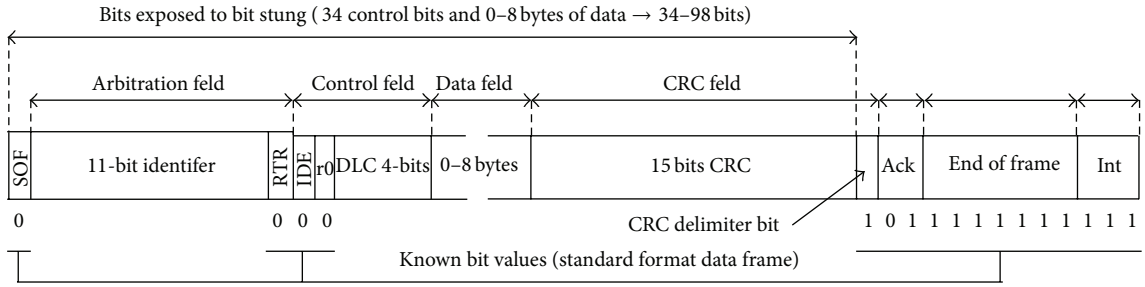


Figure 4: Standard CAN message format [17].

as well as host Central Processing Units (CPUs) and the bus. Facilitating the operation of everything from motors and braking systems to windshield wipers and door locks, CAN is currently the most popular communication standard in the automotive industry with annual sales exceeding 400 million [8] and thus provides a suitable benchmark with which we compare the JAUS standard.

ECU wired to the bus act as an interface between the bus and a local CPU which process data or initiate outgoing messages. Transmission of messages occurs at varying speeds depending on the needs of the network. Class A networks are less than 10 kb/s, Class B networks operate at 10–125 kb/s, and Class C networks range from 125 kb/s–1 Mb/s [8].

There are two versions of the protocol with the most significant difference in terms of the length of the identifier field: one protocol version has an identifier field of 11 bits, which is termed Standard CAN Version 2.0A, while the other has a length of 29 bits called Extended CAN Version 2.0B [16]. Further discussion of CAN will focus only on the Standard CAN since the benchmark analysis we will be using evaluates that format. Figure 4 illustrates the Standard CAN message format.

The maximum message size of Standard CAN messages, with no stuff bits, is 111 bits. There are 44 control bits, a maximum of 8 bytes, or 64 bits of data, and 3 interframe bits used in between message transmission.

The Standard CAN uses six bits of either all 0 s or all 1 s to indicate an error. It is problematic if a nonerror message contains a sequence of six 0 s or 1 s since the message will be viewed as an error. To overcome this, CAN utilizes a bit-stuffing scheme. During the creation of a message, if five bits in a row are all 0 s or all 1 s, the sixth bit is a stuff bit of the opposite polarity. Nodes receiving the message will automatically remove the stuff bits before processing the message. This solves the problem of differentiating between error and nonerror messages but at the cost of increased message size in certain messages. Messages with sequences of six or more consecutive 0 or 1 bits will be longer than messages with fewer, or no, sequences of six or more 0 or 1 bits due to the added stuff bits. As a result, the lengths of CAN messages are variable which means some messages will take longer than others to send.

The maximum size of a Standard CAN message, including stuff bits, can be easily calculated using equation (1) from [9], where s_m is the number of data bytes in the message. The number of control bits subject to bit stuffing is 34 and there are a total of 47 control bits:

$$8s_m + 47 + \left\lfloor \frac{34 + 8S_m}{5} \right\rfloor. \quad (1)$$

Another type of message is the Remote Transmission Request (RTR) message. This is used to signal a receiving node which has the same identifier to transmit the data that has for the identifier. This type of message is used for data which is used infrequently and does not factor into the analysis of [9].

4.2. CAN Performance Analysis of Tindell and Burns [9]. The performance of CAN is generally measured in terms of worst-case response time. This measurement begins when a task first starts to queue a message and ends when the message arrives at its destination [9]. The problem of bounding response times in a CAN system is a scheduling problem since many messages are vying to be sent, all with deadlines of when they must be received. Consequently, bounding response times in CAN is a difficult problem and a great body of work deals with the subject with varying methods and results.

For our experiments, we have selected the work of Tindell and Burns [9]. Although the schedulability analysis in [9] has been revised and replaced by that in [18], we use [9] as our benchmark reference for a couple of reasons. First, the work is cited in over 200 papers dealing with performance and scheduled in CAN. In addition, the research has influenced the design of the Motorola msCAN peripheral and the Volcano Network Architect CAN scheduled analysis tool. The second reason is that this work does not account for lost packets, error transmissions, and so forth. In our research we evaluate the simplest form of JAUS and this work provides a simple CAN analysis suitable for comparison.

The test case for this work was developed for a Class C network according to the performance guidelines in [3]. The analysis does not account for error transmissions or RTR messages. As a result, the analysis is simpler since all messages are transmitted periodically and no sporadic messages interfere with the analysis.

The worst-case response time for a message "m" is calculated using (2), where R_m is the worst-case response time for message m:

$$R_m = J_m + w_m + C_m, \quad (2)$$

where J_m is the queuing jitter for message m and is the time required to place a message in the queue. w_m is the worst-case time in which a message sits waiting in the queue for a lower-priority message to finish transmitting or for messages with a higher priority to transmit. C_m is the time required to actually send the message. So the worst-case time response is the time required to load the message into the queue, the time spent waiting to gain access to the bus, and the time to transmit the message.

We note that Table 1, contains results from the analysis of Tindell and Burns [9]. Their analysis made use of the SAE benchmark for CAN systems. In the SAE benchmark, there are 53 different messages. Some are sent periodically, while others are sporadic. For the sake of simplicity, Tindell and Burns have given all messages a period.

In the SAE benchmark, each message has a unique number to identify the message type. The deadline of each message does not correspond with the numbering. So in Table 1, while the numbering in the Signal number column is not sequential, it is in order of priority.

The messages in Table 1 are ordered in highest-to-lowest priority from top to bottom. The column "T_m" represents the period of message "m," which is the smallest time between consecutive queuing events for message m. An "x" mark indicates that a message misses its deadline.

In the original work, fifty-three messages were included. For our analysis, the first eighteen will be more than sufficient. We have defined all the parameters which appear except for the period "T_m." Tasks running on local CPUs are responsible for initiating the transmission of messages. The events which trigger a task to transmit a message do so within a time frame. The smallest difference in the time of consecutive executions of a task is the period.

We will briefly demonstrate how the values in Table 1 are obtained. We will calculate the worst-case response time of the first row under the column corresponding to 250 Kbit/s. The calculation comes directly from (2). The queuing jitter, or J_m, is not factored into the calculations in Table 1. However, the rest of the calculation remains intact.

w_m is the longest time which a message will have to wait for the wire to be clear for transmission. Since the first row corresponds to the message with the highest priority, the longest it will have to wait is for one message. This message is, in the worst case, the maximum message size. To calculate this value, we use (1), filling in 8 for s_m. This will give us the number of bits which will be sent, which is 130 bits. To calculate the time required to send these bits we simply look to the speed of the bus, 250 Kbits/s. 1 bit is sent in 1/250000 s or 4 μs. Multiplying this by 130 we find that in the worst case, the message with the highest priority will have to wait 0.00052 s or 0.52 ms.

The next calculation is C_m which is just the time needed to send the current message. Again we use (1), except that we use 1 for s_m since the size of the message is only one byte. The number of bytes in this message is 63 and multiplying this by the time required to send a single byte, 4 μs, we find the time is 0.000252 s or 0.252 ms. Adding this value to the value from the previous paragraph we find the worst-case response time for the message with the highest priority on a bus with a speed of 250 Kbits/s is 0.772 ms. The subsequent values in the column can be calculated by finding the value of C_m as we just did and adding to the worst-case response times of all of the previous messages.

5. Joint Architecture for Unmanned Systems (JAUS)

JAUS was created to be a universal UV communication standard to allow for interoperability among all UVs made by different manufacturers including avionic, marine, and ground vehicles which are compliant with the standard. The goal is for JAUS to be used for tactical functionality, such as mission planning and coordination with other UVs during the execution of missions, as well as internal communication for controlling engines, braking systems, and so forth. Current systems which are responsible for the low-level control of vehicles, such as CAN, have strict performance requirements. UVs developed for military use also have strict requirements since the UVs operate in hostile environments. In order for JAUS to be considered as a viable standard, it must be able to meet current performance standards for mission-critical vehicles.

5.1. JAUS Structure. The JAUS network is structured as a hierarchy, which is shown in Figure 5. At the top is the system, which is made up of all of the UVs which will use the network to communicate. Each UV is considered a subsystem. A subsystem is the highest element inside a UV. The software operating at the subsystem level, which processes incoming messages and forwards outgoing messages, is called the Communicator. All JAUS messages entering and exiting a UV must pass through the Communicator [19]. If JAUS messages are exchanged only between subsystems, the system is Level I Compliant. Beneath the subsystem is the node which is a computer or embedded system onboard a UV. The software operating at the node level is called the Node Manager, which is responsible for routing messages to the components it oversees and for routing outgoing messages from its components to the subsystem. Since JAUS messages are exchanged between both subsystems and nodes, the system is Level II Compliant. At the lowest level is the component which is software that directly interacts with hardware such as a motor, camera, or steering. In this case, the system is Level III Compliant since JAUS messages are exchanged between subsystems, nodes, and components.

An incoming message to a JAUS-compliant UV must first communicate with the subsystem. The message is then routed to the appropriate node, which forwards the message to the component that should receive it. When a message is sent out to another UV, the message must be routed upwards. If a component is sending a message to another subsystem, it must first send the message to its managing node. The node forwards the message to the subsystem which then sends the message to the subsystem of the intended UV. From there, the message is propagated down to the intended recipient.

Internal communication must also be routed through the hierarchy and no direct interaction between components is

Table 1: CAN performance results [9].

Signal number	Size/bytes	J/ms	T/ms	D/ms	R (125 Kbit/s)	R (250 Kbit/s)	R (500 Kbit/s)	R (1 Mbit/s)
14	1	0.1	50.0	5.0	1.544	0.772	0.386	0.193
9	1	0.2	5.0	5.0	2.048	1.024	0.512	0.256
49	1	0.2	5.0	5.0	2.552	1.276	0.638	0.319
42	1	0.2	5.0	5.0	3.056	1.528	0.764	0.382
8	1	0.1	5.0	5.0	3.560	1.780	0.890	0.445
7	1	0.1	5.0	5.0	4.064	2.032	1.016	0.508
43	1	0.1	5.0	5.0	4.568	2.284	1.142	0.571
11	1	0.1	5.0	5.0	x 5.072	2.536	1.268	0.634
32	1	0.1	5.0	5.0	x —	2.788	1.394	0.697
29	1	0.3	10.0	10.0	x 10.112	3.040	1.520	0.760
30	1	0.4	10.0	10.0	x —	3.292	1.646	0.823
53	1	1.5	50.0	20.0	x 25.232	3.544	1.772	0.886
48	1	1.4	50.0	20.0	x 29.768	3.796	1.898	0.949
46	1	1.3	50.0	20.0	x 39.344	4.048	2.024	1.012
44	1	1.2	50.0	20.0	x 39.848	4.300	2.150	1.075
40	1	1.1	50.0	20.0	x —	4.552	2.276	1.138
39	1	1.0	50.0	20.0	x —	4.804	2.402	1.201
27	1	0.9	50.0	20.0	x —	7.072	2.528	1.264
38	1	0.9	50.0	20.0	x —	7.324	2.654	1.327
37	1	0.8	50.0	20.0	x —	7.576	2.780	1.390
52	1	0.8	50.0	20.0	x —	7.828	2.906	1.453
26	1	0.8	50.0	20.0	x —	8.080	3.032	1.516
35	1	0.7	50.0	20.0	x —	8.332	3.158	1.579
51	1	0.7	50.0	20.0	x —	8.584	3.284	1.642
22	1	0.7	50.0	20.0	x —	8.836	3.410	1.705
34	1	0.6	50.0	20.0	x —	9.088	3.536	1.768
20	1	0.6	50.0	20.0	x —	9.340	3.662	1.831
50	1	0.6	50.0	20.0	x —	9.592	3.788	1.894
31	1	0.5	50.0	20.0	x —	9.844	3.914	1.957
47	1	0.5	50.0	20.0	x —	12.616	4.040	2.020
28	1	0.5	50.0	20.0	x —	12.868	4.166	2.083
19	1	0.5	50.0	20.0	x —	13.120	4.292	2.146
25	1	0.4	50.0	20.0	x —	13.372	4.418	2.209
17	1	0.4	50.0	20.0	x —	13.624	4.544	2.272
45	1	0.4	50.0	20.0	x —	13.876	4.670	2.335
24	1	0.3	50.0	20.0	x —	14.128	4.796	2.398
16	1	0.3	50.0	20.0	x —	14.380	4.922	2.461
18	1	0.3	50.0	20.0	x —	14.632	6.056	2.524
41	1	0.3	50.0	20.0	x —	14.884	6.182	2.587
23	1	0.2	50.0	20.0	x —	17.152	6.308	2.650
15	1	0.2	50.0	20.0	x —	17.404	6.434	2.713
6	1	0.9	100.0	100.0	x —	17.656	6.560	2.776
4	1	0.8	100.0	100.0	x —	17.908	6.686	2.839
2	1	0.7	100.0	100.0	x —	18.160	6.812	2.902
1	1	0.6	100.0	100.0	x —	18.412	6.938	2.965
12	1	0.4	100.0	100.0	x —	18.664	7.064	3.028
10	1	0.2	100.0	100.0	x —	18.916	7.190	3.091
36	1	1.7	1000.0	1000.0	x —	19.168	7.316	3.154
33	1	1.6	1000.0	1000.0	x —	19.420	7.442	3.217
13	1	1.2	1000.0	1000.0	x —	19.672	7.568	3.280
5	1	1.1	1000.0	1000.0	x —	22.444	7.694	3.343
3	1	1.0	1000.0	1000.0	x —	22.696	7.820	3.406
21	1	0.3	1000.0	1000.0	x —	22.948	7.946	3.469

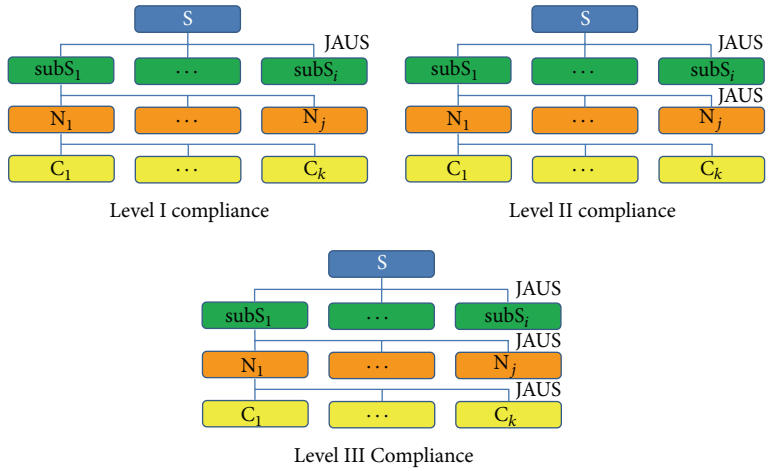


Figure 5: JAUS hierarchy and Compliance Level. S: system, subs: subsystem, N: node, C: component.

allowed. For instance, if a component belonging to a node needs to communicate with another component belonging to the same node, the message must first go through the node. Similarly, messages going from one node to another in the same subsystem must first be routed through the subsystem. This hierarchical structure for message routing in Figure 5, as opposed to direct communication between components, contributes in part to the performance issues of JAUS [2] and is something we will address in our experiments.

All components, including the Communicator and Node Manager, in a JAUS-compliant system are required to have a unique name and JAUS IP address [19]. The name can be an upper- or lower-case letter, a number, or an underscore. The JAUS IP is a dotted-decimal value, the same as a standard IP address. The only difference is that JAUS IPs are used internally to identify JAUS components and generally cannot resolve to an IP address.

5.2. JAUS Messages. There are currently one hundred fifty-seven JAUS messages [20]. The JAUS message header is 16 bytes of data. The destination and source of JAUS messages are encoded in the header as four bytes each, one byte for each octet. It also contains a 16-bit field, the Command Control field, which designates the type of message, and a field for data length, among others. The size of JAUS messages is variable, restricted only by the frame size of the protocol in which the message is encapsulated. The JAUS header is depicted in Figure 6.

JAUS currently allows for "experimental" messages. Developers who feel unable to perform all necessary functionality with the current one hundred fifty-seven messages can create their own messages within the experimental message range. Experimental messages are available but should be used sparingly as they are not part of the standard and interoperability between JAUS-compliant UVs may suffer if experimental messages are present in some and not others.

JAUS currently has not defined a communication protocol. It is believed, however, that UDP and RS-232 will be claimed as the protocols of choice for JAUS [22]. Additionally, transmission of audio/video data is not yet supported [21] and the specifics of mission planning have yet to be defined. Another very important facet the JAUS standard has yet to address is performance.

5.3. JAUS Communication. One of the principal goals of JAUS is to provide a level of interoperability between intelligent systems that has been missing in the past. Towards this end, JAUS defines functional components with supporting messages but does not impose regulations on the systems engineer that govern configuration.

JAUS does have one absolute, unwavering requirement that can have an effect on configuration. To achieve the desired level of interoperability between intelligent computing entities, all messages that pass between JAUS defined components (over networks or via airwaves) shall be JAUS compatible messages [19]. In order to present a JAUS communication, we provide a diagram shown in Figure 7.

This figure depicts two JAUS defined components in the right two big boxes and a dedicated hardware device in the left small box. An explanation of the figure is given below.

(1) Node 1 is configured as a positioning unit and is dominated by the Global Pose Sensor component. The Differential Global Positioning System (DGPS) is configured as a dedicated device within the node. The DGPS is setup in streaming mode so that positioning data is sent out over Recommended Standard 232 (RS-232) at some regular interval.

(2) Node 2 is configured as a Path Driving unit and is dominated by the Global Path Segment Driver component. It relies on Global Pose Sensor messages to perform its path-tracking task.

(3) The Global Pose Sensor component task reads shared memory whenever necessary to get the latest DGPS information. (Important note: so far all of the data

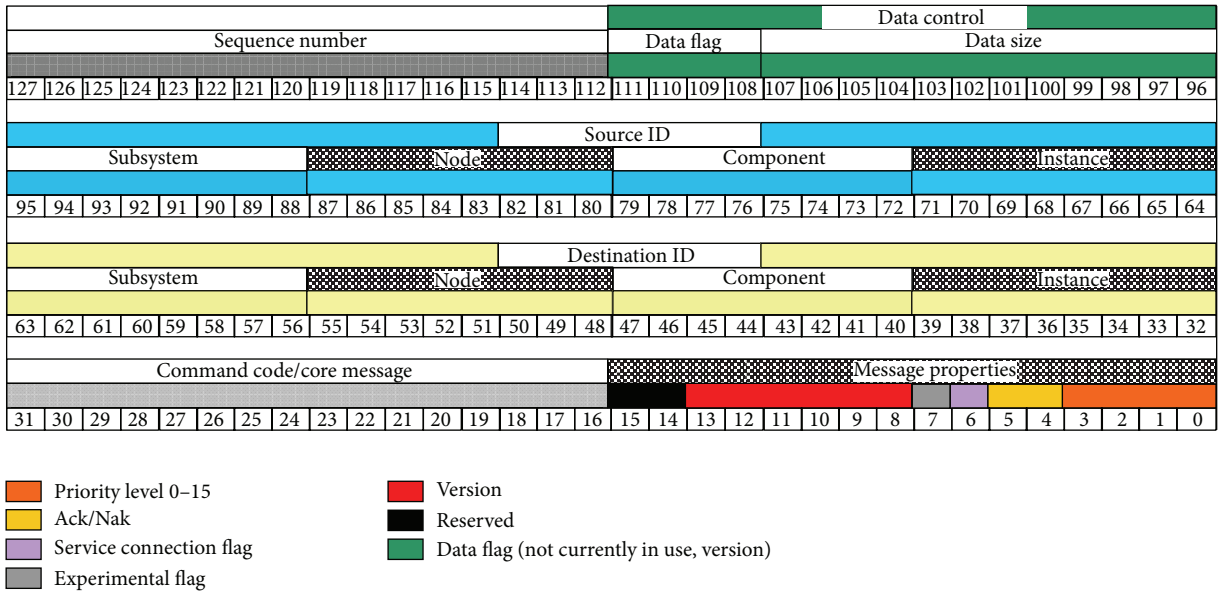


Figure 6: JAUS message header [21].

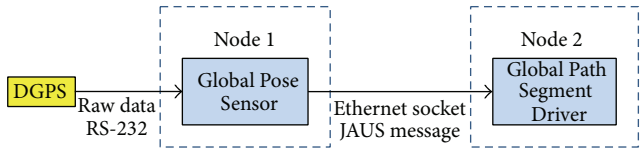


Figure 7: JAUS communication example configuration.

is in raw (non-JAUS) format because typically dedicated sensors, like the DGPS, do not support JAUS messages.)

(4) When the Global Pose Sensor component completes its position calculations, it sends a JAUS formatted Global Pose Sensor message to Node 2. This sequence loops continually. Notice that although the streaming data coming from the DGPS to the I/O server and on to shared memory is not in JAUS format, the interoperability rule remains unbroken. Only when data is sent between JAUS defined components, must it be formatted into a JAUS compatible message.

6. JAUS Performance Evaluation

6.1. JAUS Implementation. For our experiments, we created a simple JAUS implementation focusing on the kernel, or core part, of JAUS, called "k-OpenJAUSC++," written in C++ for the Windows operating system. The project contains two subprojects: The "Lib" subproject and the "Communicator" subproject. The "Lib" subproject contains classes dealing with constructing and processing messages and is static. Once the message creation and processing functions have been created, there is really no need to modify them. Since the majority of work occurs in the "Communicator," we compile it as a Dynamic Link Library (dll) to simplify the Communicator. The "Communicator" is much more dynamic, responsible obtaining the configuration of the JAUS system, running the server, and processing messages. The class diagram for the "Communicator" and "Lib" and the sequence diagram for the "Communicator" receiving a message appear in Figures 8, 9, and 10, respectively.

It is already known that existing, full-featured JAUS implementations have performance issues. The question that arises is what causes these performance issues. We have opted to create a simple JAUS implementation in order to evaluate the performance of the JAUS standard at its core. By testing the performance on a simple JAUS implementation, determining the causes of the performance is much simpler. We believe that the main cause of latency in the JAUS standard is due to the hierarchical message passing scheme of JAUS coupled with the use of UDP as the transport protocol. As such, our JAUS implementation focuses on creating a JAUS message, sending it using the UDP protocol in compliance with the JAUS standard, and processing the message when it is received. This allows us to more easily evaluate the effects of the hierarchical JAUS message passing scheme and the use of UDP as the protocol. Only one application is necessary for use as subsystems, nodes, and components. Each application has a multithreaded server which is capable of creating and sending, routing, or processing JAUS messages. If a message is received that is intended for another subsystem, node, or component, it is forwarded to the next element in the path to the destination. If a message is received by the intended recipient, the message is processed. As a result, our code is reusable, requiring minimal modifications only to instances which need to send or receive a message.

The network of the JAUS system is defined in an XML file. Each subsystem, node, and component application that is running has a copy of the XML file so it is able to determine where to send JAUS messages. This requires

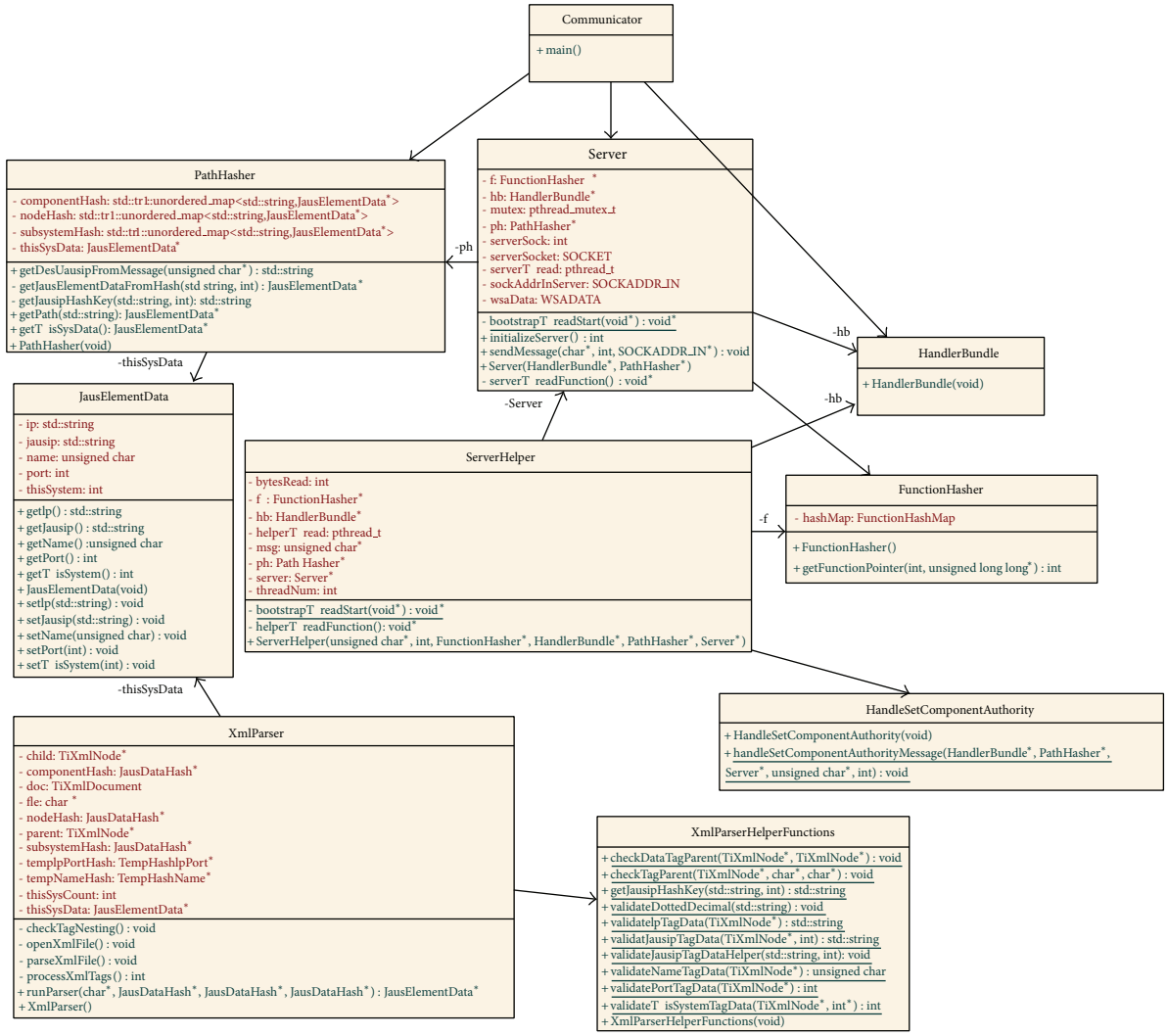


Figure 8: Class Diagram for Communicator.

the entire JAUS system to be mapped out ahead of time. The JAUS standard does allow for broadcasting for discover of components, but for simplicity, we have opted to use static system configurations. The XML file contains the JAUS name, IP address, receiving port, and JAUS IP address of each subsystem, node, and component that is in the system. Additionally, a fifth tag labeled "thissystem" contains a value of 1 to indicate the information for the current application and 0 for all others. Algorithm 1 contains a sample XML file used in our implementation.

When our implementation begins, the "Communicator" is the entry point into the application. This calls the "PathHasher," which is responsible for storing the system information from the XML file, the Server, which listens for incoming messages, and the "HandlerBundle," which is used for passing data, API access, and so forth, to the functions which process messages.

The "XMLParser" class is responsible for extracting the data from the configuration XML file. For the core XML parsing capabilities, we used the open-source TinyXML [23] code. The data for each subsystem, node, and component is stored in a "JausElementData" instance. The "XMLParser" provides the "PathHasher" with a pointer to the "JausElementData" instance which holds the data for itself to allow for each application to quickly determine its own data.

All of the "JausElementData" instances are stored in one of three unordered maps, or hash maps, one for subsystems, one for nodes, and the third for components. The "PathHasher" maintains these three hash maps for use in determining where a JAUS message should be sent next to arrival at its destination in compliance with the JAUS standard.

The "PathHasher" resolves where to route messages in the following way. The hash keys resolve to an instance of the "JausElementData" class which contains the data for a single subsystem, node, or component specified in the XML file. In the hash map for the subsystems, the key is the first octet in the destination JAUS IP as a string. For the nodes, the key value is the first octets in the JAUS IP, including the decimal

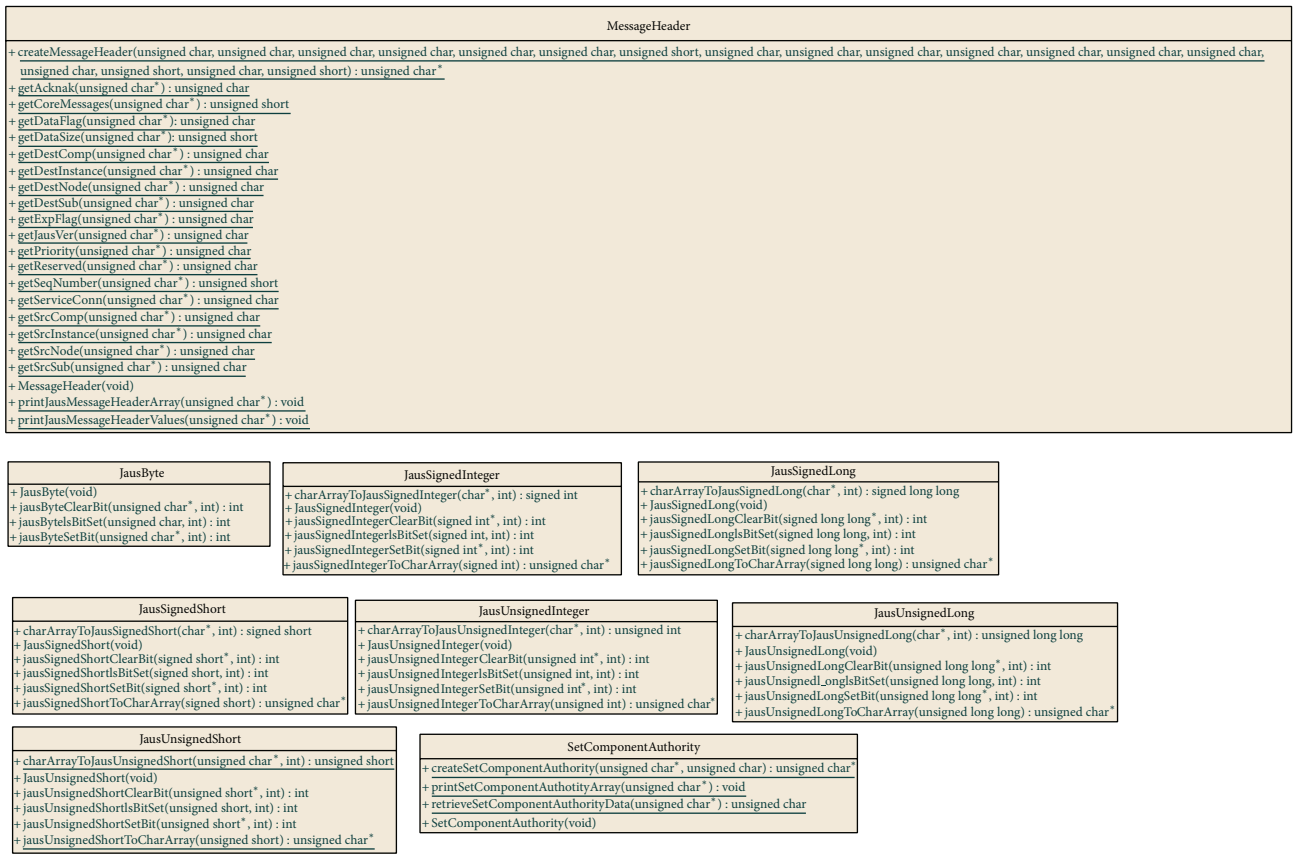


Figure 9: Class Diagram for Lib.

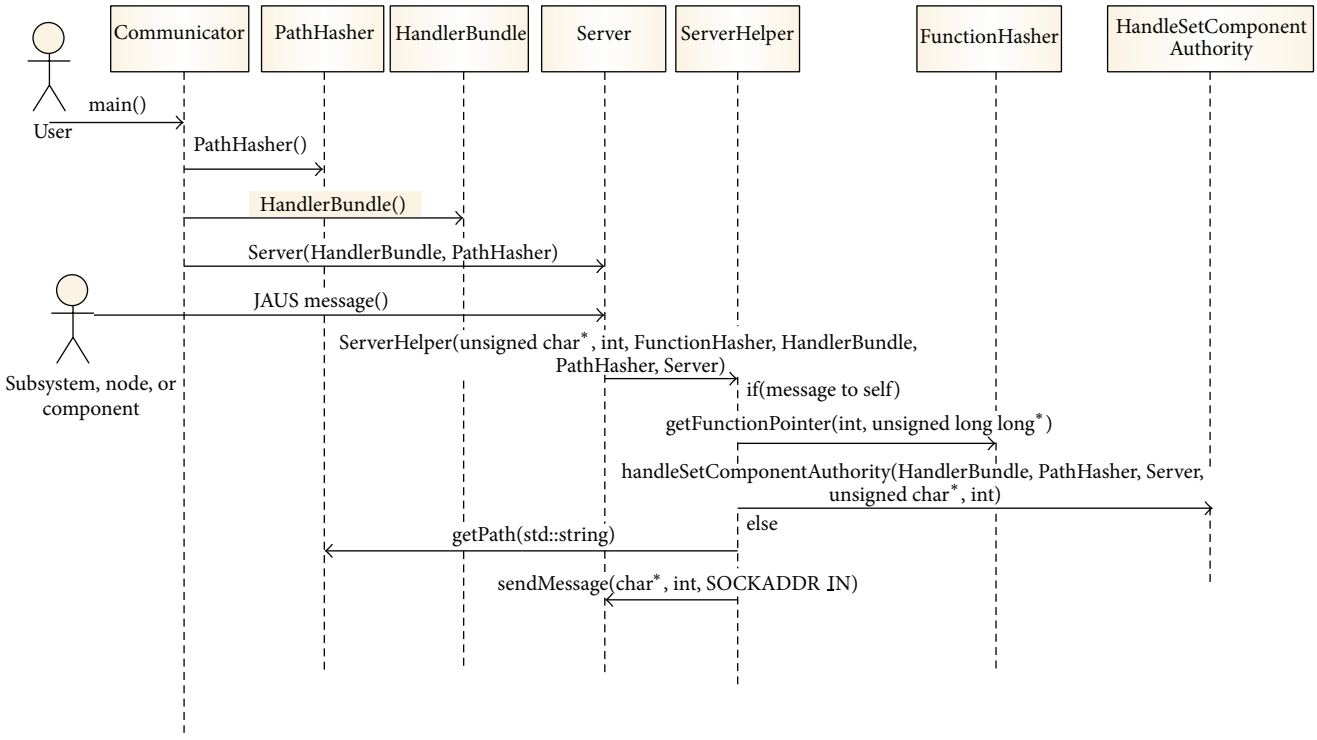


Figure 10: Sequence diagram of Communicator receiving a message.

```
<?xml version="1.0"?>
<system>
        <subsystem>
                <name>a</name>
                <ip>128.208.244.165</ip>
                <jausip>1.0.0.0</jausip>
                <port>39777</port>
                <thissystem>1</thissystem>
        </subsystem>
        <subsystem>
                <name>b</name>
                <ip>128.208.244.229</ip>
                <jausip>2.0.0.0</jausip>
                <port>39777</port>
                <thissystem>0</thissystem>
        </subsystem>
</system>
```

Algorithm 1: JAUS configuration file.

point, as a string. For the components, the key value is the JAUS IP, in its entirety, as a string.

To see why this works, consider a node which receives a message. The node must determine whether to send the message to its subsystem, process the message, or forward the message to one of the components it oversees. The node first checks if the JAUS IP in the message matches its own JAUS IP. If not, it compares the first two numeric values of its JAUS IP to the first two numeric values of the JAUS IP the message it is intended for. If they match, the message is intended for a component which the node oversees. The node uses the destination JAUS IP from the message as a hash key in the hash map for the components to get the IP address and port that it should forward the message to. If the values do not match, the node must pass the message to its subsystem; it uses the first numeric value of its own JAUS IP as the hash key in the hash map for the subsystems. By using hash maps, routing within JAUS systems can be efficiently handled.

At each point along the path a JAUS message takes from its origin to its destination, there are a few validation steps that all applications take when a message is received. The message is checked to determine that its size is at least the size of the header, 16 bytes, the JAUS version byte is checked to determine that it is the correct JAUS version and the length of the data portion of the JAUS message is compared to the message size field in the header to ensure that they match. For our simple JAUS implementation, this is sufficient validation.

When a message is received by the intended application instance, the message must be processed. Each message type has a handler class which contains the code to process the message. The JAUS standard, as of Version 3.3, has one hundred fifty-seven messages [21] and performing as many if else statements to determine the message type could negatively impact performance. Hash maps are used to make the process of resolving an incoming message to the class which should process it more efficiently. In the command core field of the JAUS header is a numerical value which indicates the type of message. This integer value is used as a key to a hash map which resolves to a pointer to the static function which processes the message. The number of message types is static, which means that during the execution of the program no values are added or removed from the hash map, which means the performance of the hash map will not degrade. This allows the application to determine, in constant time, the proper function to process each message type.

The "FunctionHasher" resolves to a specific function in a handler class for each type of message. When a handler class is processing a message, it may need access to external variables, classes, and so forth. For instance, if a message is received which must alter the speed of the engine in a UV, the handler class must have access to the API for engine control. The "HandlerBundle" class deals with this problem. This class is passed to all message handling classes which can use any classes, variables, or data included in "HandlerBundle." This way, developers can avoid altering the core functionality of the JAUS implementation. All that needs to be done is to include the necessary elements in the "HandlerBundle" class and then code each message handling class to process messages in the appropriate manner.

6.2. JAUS Experiments. We conducted eight different experiments to assess the performance of the JAUS standard when using the UDP protocol. All of the experiments test the time required to construct, send, receive, and process a JAUS message in order to determine worst-case response-time results. We used a single message type for all of our tests, the "Set Component Authority" JAUS message. The message has the standard 16-byte JAUS header and the payload is a single byte. The reason for selecting this message type is that it is similar to a CAN message in size. The worst-case response-time analysis for CAN generally uses the maximum message size of 132 bits. The "Set Component Authority" message is 17 bytes, or 136 bits, a trivial difference from the CAN message. This allows us to rule out message size as a significant

source of latency in the JAUS system. The tests are listed below.

(1) Test the time required to send a JAUS message from a component in one subsystem to a component in another subsystem.

(2) Test the time required to send a JAUS message from a node in one subsystem to a node in another subsystem.

(3) Test the time required to send a JAUS message from a subsystem to another subsystem.

(4) Test the time required to send a JAUS message from a component to another where both components are in the same subsystem and share the same node.

(5) Test the time required to send a JAUS message from a component to another where both components are in the same subsystem but have different nodes.

(6) Test the time required to send a JAUS message from a component to its node.

(7) Test the time required to send a JAUS message from a node to its subsystem.

(8) Test the time required to send a JAUS message from a component to its subsystem.

For all of the experiments, a subsystem and its nodes and components are all on the same computer. The first three experiments require the use of two different computers since the message passes between different subsystems at some point during each of these experiments. This allows us to examine how performance is affected by messages which must pass through a router to reach their final destination in addition to the effect that longer paths through the JAUS hierarchy have. For the last five experiments, only a single computer is needed since all of these experiments take place within a single subsystem. Tests 4 and 5 are complete JAUS transactions in full compliance with the JAUS standard which allows us to determine how shorter and longer paths through the JAUS hierarchy perform. Tests 6 and 7 allow us to gauge the speed of the UDP protocol since they both test direct communication between elements in a JAUS system. Test 8 gives us the time necessary for a message to get to a component once it has entered a UV through the subsystem.

In the experiments, a subsystem and its nodes and components are all kept on the same computer for a more favorable measure of time. Current research has indicated that the performance of JAUS is too slow when using the UDP protocol and we are operating under that assumption [2]. If the elements in the subsystems are on separate computers, the transmission time will be longer since the message will have to travel through wires and routers at each point to reach its destination. Transmitting messages between elements which are on the same computer requires less time since the messages stay on the same motherboard. This provides a best-case performance for the experiments and for the performance of JAUS. A favorable outcome of our experiments when compared with the CAN results indicates further research is necessary to pinpoint the source of latency in JAUS. If, even under favorable conditions, the performance of JAUS cannot compete with CAN, we have found the source of latency.

For each experiment, we would like to determine the time required for one subsystem, node, or component in a JAUS system to reach another subsystem, node, or component elsewhere in the same JAUS system. In the first three experiments, the sending and receiving elements are on different computers, so getting the start time on the first computer and the finish time on the second computer may not provide accurate results due to the clocks in the computers not being synchronized. To overcome this, we determine the round trip time and halve it.

Each experiment will test the transmission time of 1000 messages so that we have an adequate sample from which a reasonable transfer time can be inferred. A "for" loop, counting from 0 to 999, inclusive, is used to send and track the tests. The sending subsystem, node, or component, which we will refer to as the Initiator, will get the current time and save the value in an array using the value from the "for" loop as the index to which the value is saved. Then, the JAUS message is constructed with the sequence number field containing the index value from the "for" loop. The message is then sent and routed through the JAUS hierarchy, in compliance with the standard, to its destination, which we will refer to as the Receiver. The Receiver processes the message and retrieves the index value from the sequence number field. It then constructs its own JAUS message and sets the sequence number in the message to the value of the sequence number in the message which it received from the "Initiator." The "Receiver" then sends its message back to the "Initiator." When the "Initiator" receives this message, it processes it and retrieves the sequence number. The current time is then saved in the array in the index corresponding to the sequence number from the message. This process occurs 1000 times, with a 100 ms pause between each message sent by the "Initiator."

From the data we glean four statistics: the minimum transmission time, the maximum transmission time, the average transmission time, and the standard deviation. The difference in the start and finish time is calculated for each message and divided by two to get a reasonable estimate of the transmission time of a one-way trip for the JAUS messages. UDP is a connectionless protocol so packets may be lost during transmission. The array containing the time values is zeroed out prior to each message being sent. When calculating the statistics, the application checks if the return time is 0, which indicates that a packet was lost during the test. The number of tests used as the divisor in both the average and standard deviation calculations is accordingly adjusted to not include lost packets. Tests four through eight are all only on one machine, which means we could run the tests by simply taking the start time at the "Initiator" and the finish time at the "Receiver." We opted to perform all of the tests in the same manner to maintain consistency.

The time measurement tool which is used to measure the elapsed time is the "QueryPerformanceCounter" tool provided by Microsoft in Visual Studio 2008. The "QueryPerformanceCounter" provides time data in the form of clock

cycles per second since the computer was started. To obtain the time in seconds, the number of clock cycles the computer performs each second is divided into the time value.

For our experiments, we used two identical computers running the Windows XP Service Pack 3 operating system. The chips are the Intel 2 Quad Core running at 2.40 GHz and the ram size is 3.0 GB.

6.3. Results and Analysis. Our analysis highlights a few different problems causing performance issues with JAUS. The hierarchical message passing scheme required by JAUS is one source of latency. Directing messages through subsystems and nodes, instead of directly between sending and receiving components, introduces latency into the system. Another issue is the use of the UDP protocol. In addition to being slower than CAN systems with fast bus speeds, UDP provides no scheduling mechanism. The results of our experiments are provided in Table 2 and a graph in Figure 11.

6.3.1. JAUS Hierarchy. We first examine how the hierarchical routing scheme of JAUS affects performance. In Figure 11, the results of each of the eight tests are plotted. In addition to the mean, the standard deviations above and below the mean are also plotted to give a sense of the variation in transmission times. As hypothesized, the more the subsystems, nodes, and components which a message must traverse to reach its destination are, the greater the time is.

Transmitting from component to component in Test 1 requires the greatest time, transmitting from node to node takes less time in Test 2, and transmitting directly between subsystems in Test 3 takes the least amount of time of these three tests. Tests 4 and 5 test transmission times between two components in the same subsystem. In Test 4, both components belong to the same node while in Test 5 the components belong to different nodes. The message in Test 4 only needs to pass from the sending component to the node before reaching its destination at the receiving component. Test 5, however, requires the message to pass from the component all the way to the subsystem before it can be sent to the receiving component by way of its managing node, and the difference in performance is noticeable. Test 5 performs more poorly than Test 1 even though Test 1 must go through a router to bridge the gap between the two machines.

Tests 6, 7, and 8 test the necessary time to transmit a JAUS message between a component and node, a node and subsystem, and component and subsystem, respectively. Tests 6 and 7 are very similar since they both measure the time required for direct communication between elements in the JAUS hierarchy. Test 8 requires an additional hop due to the node and as a result underperforms in comparison to Tests 6 and 7.

The indirect message routing scheme of JAUS has a noticeable impact on performance as expected. We now use this data in the context of Table 1. The results in Table 2 detail the time required to send a minimal JAUS message for each of our tests. In Table 1, the first nine messages have a deadline of 5.0 ms. Using the average response time from Table 2, we see that in Tests 1, 2, 5, and 8 the transmission speed is too slow;

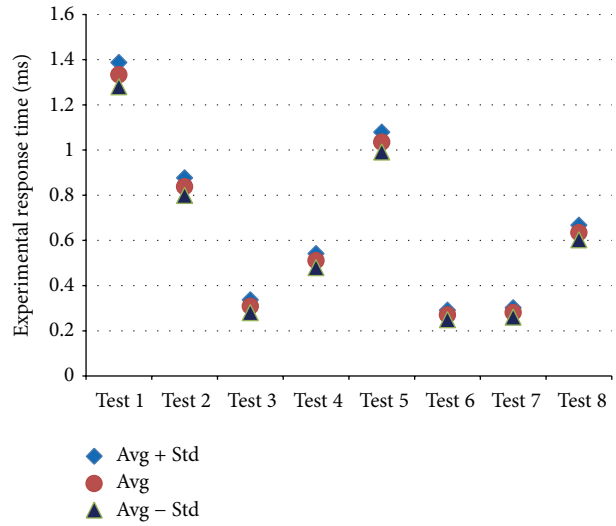


Figure 11: JAUS experiment test results.

Table 2: Experiment results (the time is in the units of ms).

Test number	Minimum response time	Maximum response time	Average response time	Standard deviation
1	0.417706	1.643011	1.333256	0.053709
2	0.417706	1.099552	0.837777	0.038997
3	0.275044	1.045092	0.308805	0.028363
4	0.417706	0.656319	0.510614	0.031094
5	0.417706	1.303495	1.035215	0.043867
6	0.227896	0.392914	0.270172	0.021158
7	0.234931	0.381488	0.281037	0.020506
8	0.417706	0.819728	0.634676	0.032273

if nine JAUS messages were sent consecutively, each with a deadline of 5 ms, not all of the messages would make their deadlines.

The configurations that are too slow have a significant amount of overhead, traveling through many nodes and subsystems. The solution, then, is to eliminate nodes and have all internal communication happening directly between nodes. This, however, is not enough, which will be demonstrated in the next section.

6.3.2. JAUS/UDP. To compare JAUS/UDP with CAN, we will ignore the JAUS hierarchy and use the results from Test 6, which tests the communication between a component and its node, a direct connection, which is also the smallest value from our results.

In Table 1, the first message is the highest priority message and its response time is representative of the response time in the CAN network of a message that is one byte in size. Since all of the messages in Table 1 are one byte, we can use the response time for the first message as a reasonable indicator of the speed of 1-byte messages in a CAN system. If we compare the response times for the first message in

250 Kbit/s, 500 Kbit/s, and 1 Mbit/s CAN systems with the response time for Test 6, we see that the JAUS message is actually faster than the 250 Kbit/s and 500 Kbit/s CAN systems. The problem is that raw speed is not the only problem with JAUS.

Consider the scenario in Table 1 again and use the average response time from our results in Test 6. The first nine messages have deadlines of 5 ms. The time to transmit a JAUS message from Test 6 is 0.270172 ms, so the time required to send the first nine messages is 2.431548 ms.

The problem that arises is a scheduling problem. In a scenario where all of the types of messages are being sent simultaneously, collisions will occur. As discussed previously, CAN is able to send the messages in order of priority even when collisions occur. This is not the case with UDP. If a collision occurs, all transmitting nodes stop, wait a random time period, and then attempt to retransmit. This makes it virtually impossible to know the order in which the messages will be transmitted. It is then possible that any of the ninth highest priority messages will not be one of the first nine messages sent. Any of the messages with deadlines of 5 ms must be one of the first 18 messages sent or they will miss their deadline. Messages with deadlines of 10 ms must be one of the first thirty-seven messages sent or they will miss their deadlines. Using JAUS and UDP it is possible that using JAUS and UDP all messages in the scenario in Table 1 will meet their deadlines. It is also possible that they will not make their deadlines and there is no way to guarantee which situation will occur. Obviously this level of uncertainty is not acceptable in mission-critical systems and thus JAUS/UDP cannot be used for internal controls of UVs.

7. JAUS/UDP to EtherCAT Bridge: JEBridge

To improve the performance of JAUS/UDP communication discussed in Section 6, we utilize EtherCAT as a solution, due to the fact that EtherCAT is an emerging, but promising, lower level protocol that has a lot of support. There is a general way to translate between JAUS and EtherCAT. A survey of available EtherCAT modules shows this to be impossible. EtherCAT modules have widely different device profiles. So the top down method, starting with JAUS and translating to EtherCAT, will not work. However, a bottom-up method, starting with a number of EtherCAT modules and deriving the translation, should be quite achievable.

The first question to ask when bringing EtherCAT into JAUS is where we should place the EtherCAT. In other words, at what level should we need to implement the EtherCAT? As mentioned before, motion control requires deterministic timing with bounded latency. So our approach is to implement EtherCAT at Level 3, where the EtherCAT bridge is inserted into components, and EtherCAT communication happens between nodes/subsystems. Figure 12 shows the insertion of EtherCAT into existing JAUS communication system.

In the following subsections, we will discuss the implementation of our JAUS-EtherCAT bridge.

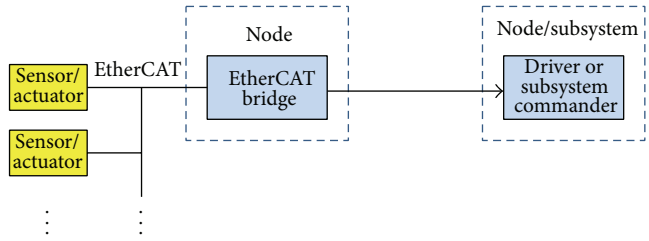


Figure 12: The next stage in the evolution to EtherCAT based sensors and actuators (Figure 3-4 in [19]).

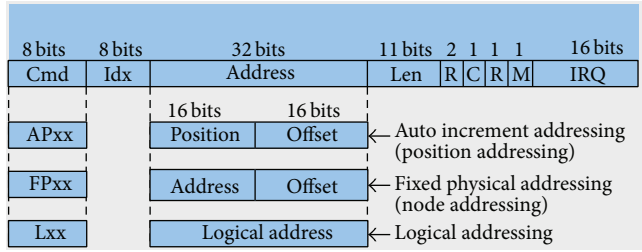


Figure 13: Address Field (slide 34 of [7]).

7.1. EtherCAT Addressing Schemes. EtherCAT modules fall into two categories: "native EtherCAT" and "transition modules" which use EtherCAT to encapsulate a legacy protocol, such as CANopen.

7.1.1. Native EtherCAT Module. We first consider native EtherCAT module. The important parts of the EtherCAT telegram header are the command (8 bits), the address (32 bits), and the telegram data size (11 bits). The EtherCAT command defines the way the address is evaluated by the EtherCAT slave devices. It may either be interpreted as a physical address (16 bits) with an offset (16 bits) within the address space of the EtherCAT Slave Controller (ESC) or as a logical address (32 bits) of a 4 GB virtual address space, and this is shown in Figure 13.

JAUS addressing consists of four fields: Subsystem, Node, Component, and Instance. Each field consists of 8 bits. Thus, the JAUS address is 32 bits total, the same size as the EtherCAT address. A higher level system engineer can easily adopt JAUS convention and philosophy as she assigns addresses to EtherCAT modules. The native EtherCAT-to-JAUS bridge is shown in Figure 14. Note that both JAUS and EtherCAT favor little endian integer representation. So in the above diagram Subsystem is the most significant byte, and Instance is the least significant byte, as convention would suggest.

7.1.2. Transition Module. Next we consider "transition module," in particular CANopen. CANopen device has to implement certain standard features in its controlling software: application (top layer), object dictionary (middle layer), and communication (bottom layer) shown in Figure 15.

The application part of the device actually performs the desired function of the device, after the state machine is set to the operational state. The application is configured by

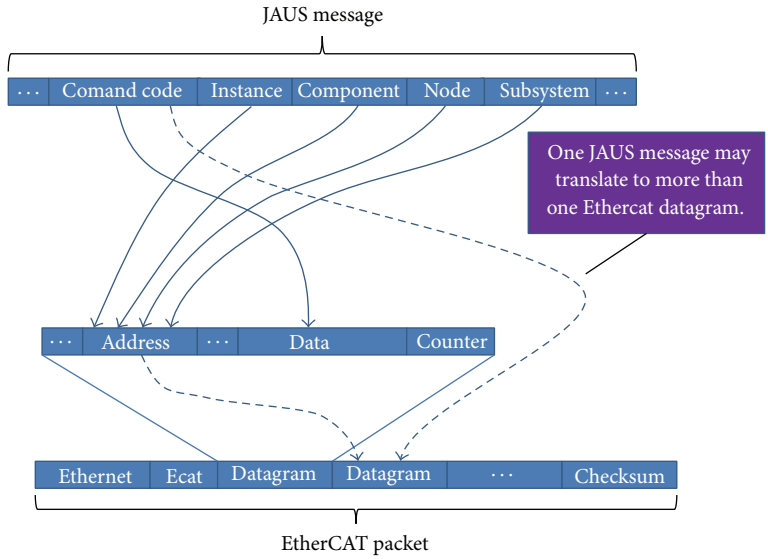


Figure 14: Address and data translation from JAUS to native EtherCAT.

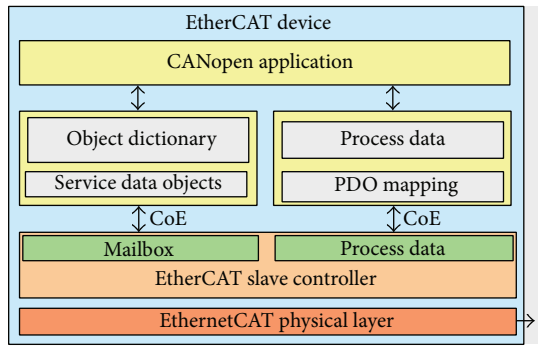


Figure 15: CANopen over EtherCAT device architecture (slide 128 of [7]).

variables in the object dictionary and the data are sent and received through the communication layer.

The object dictionary is an array of variables with a 16-bit index. Additionally, each variable can have an 8-bit subindex. The variables can be used to configure the device and reflect its environment, that is, containing measurement data.

The object dictionary is associated with Service Data Objects (SDO) protocol. The SDO protocol is used to set and read values from the object dictionary of a remote device. The device whose object dictionary is accessed is the SDO server and the device accessing the remote device is the SDO client. The communication is always initiated by the SDO client. In CANopen terminology, communication is viewed from the SDO server, so that a read from an object dictionary results in an SDO upload and a write to dictionary is an SDO download.

A communication unit implements the protocols for messaging with the other nodes in the network. Starting and resetting the device are controlled via a state machine. It must contain the states Initialization, Preoperational, Operational, and Stopped. The transitions between states are made by issuing a network management (NMT) communication object to the device.

In transition modules, the addressing scheme is the same, except that the address space for CAN is smaller; thus, some fields in the JAUS addressing scheme are unused. The transition modules to JAUS is shown in Figure 16.

7.2. Implementation of Native EtherCAT Module. We start the implement of the native EtherCAT module by defining the Ethernet and EtherCAT frame structure (refer to Figure 2 in Section 3) in a header file and then develop a function that initializes Ethernet header and EtherCAT frame header Type 1 (EtherCAT Device Communication). Ethernet header is initialized with values of Destination, Source, and Type.

We now want to send a Logical Write (LWR) packet. We first need to receive the JAUS message which is converted into a generic JAUS message. We then initialize the Ethernet header and the EtherCAT frame header. The EtherCAT datagram header is initialized with "Cmd" equaling LWR. Now, we want to implement the mapping shown in Figure 14. The JAUS destination addresses are packed and stored into the "Address" field of the EtherCAT. As mentioned earlier, each field of those JAUS "Destination ID" is 8 bits. Thus, the JAUS address is 32 bits total, the same size as the

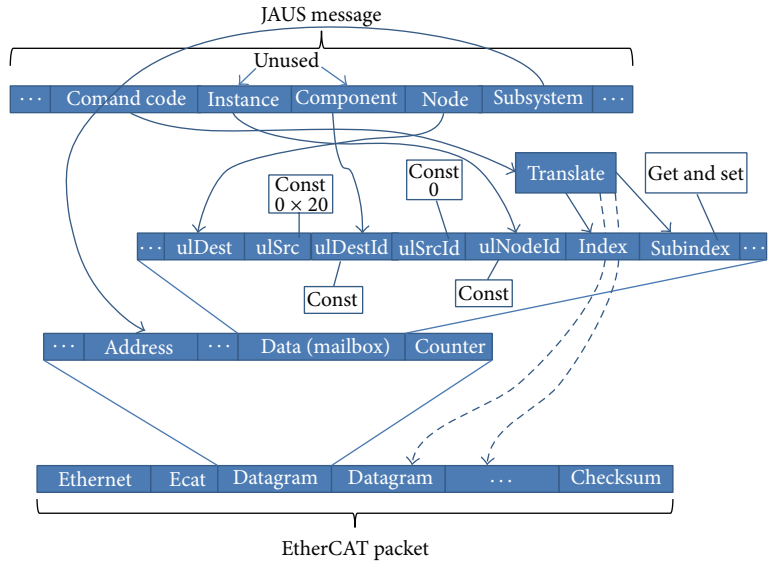


Figure 16: Address and data translation from JAUS to CoE (CAN over EtherCAT).

EtherCAT address. The packet data, which in this case an array of unsigned char, is then filled out with JAUS "SetWrenchEffort" data (as an example). The total packet length is then calculated.

The "CommandCode" of JAUS could translate into multiple EtherCAT datagram, but this is not implemented in our code. Here is the explanation for that specific mapping. Our example was for component which implemented all three axes. It is conceivable that we have three components each of which controls an axis, and in that case the single JAUS message that would be translated into three EtherCAT datagrams; and the resulting EtherCAT data from each axis would be put back into a single JAUS single message.

7.3. Implementation of Transition EtherCAT Module. We start the implement of the transition module by structure shown in Figure 17. The detailed definition and implementation of this frame is shown in Algorithm 2.

Now we are ready to send a CANopen packet named "SDO_Services_Packet" over EtherCAT. Our mapping described below is according to Figure 16.

(1) The first three mappings are mapping based on JAUS-CAN hierarchical structure. JAUS hierarchical structure is clearly defined in JAUS RA [19–21]. CAN structure is not defined in any CAN document. But we can claim that CAN hierarchy consists of Dest, DestId, and NodeId. Our justification for this claim is based upon CANopen Master Protocol [24]. Thus, JAUS Node, Component, and Instance are naturally mapped into CAN Dest, DestId, and NodeId, respectively. Note that JAUS Subsystem is not mapped into anything, because CAN already knows which Subsystem the CAN device is for.

(2) "ulSrc" of EtherCAT is filled with a constant value of "0x20" being consistent with [24].

(3) "ulSrcId" is filled with a constant value of "0" being consistent with [24].

(4) "ulIndex" takes the highest 8 bits of the JAUS "CommandCode."

(5) "ulSubIndex" takes the lowest 8 bits of the JAUS "CommandCode." In "ulSubIndex," there are two possible values: get and set (0 and 1).

The "Index" of CAN could translate into multiple EtherCAT datagram, but this is not implemented in our code. Here is the explanation for that specific mapping. Our example was for component which implemented all three axes. It is conceivable that we have three components each of which controls an axis, and in that case the single JAUS message that would be translated into three EtherCAT datagram; and the resulting EtherCAT data from each axis would be put back into a single JAUS single message. Codes for the implementation of our mapping are skipped here but we would be happy to share upon your request.

7.4. Quantitative Analysis: Performance. JAUS messages are short, and short messages over an Ethernet channel are very inefficient [25], as shown in Figure 18.

7.5. Qualitative Analysis: Nondeterministic versus Deterministic Control. JAUS is designed to be independent of communication protocol; however, the overwhelming protocol choice appears to be JAUS over UDP. Thus, our observations apply to this case. UDP has both advantages and disadvantages. The advantages include commodity hardware and quick prototyping (plug and play). The disadvantages include nondeterministic timing. UDP is a peer-to-peer, CDMA (Collision Detect Multiple Access) protocol. When a collision occurs, the packet is lost. EtherCAT is a master-slave protocol, so collisions should not occur and timing is

```
struct CAN_Packet_Header
{
   UINT32        ulDest;
   UINT32        ulSrc;
   UINT32        ulDestId;
   UINT32        ulSrcId;
   UINT32        ulLen;
   UINT32        ulId;
   UINT32        ulSta;
   UINT32        ulCmd;
   UINT32        ulExt;
   UINT32        ulRout;
};

struct CAN_Packet_Body
{
   //MbxHeader
   UINT16        Length
   UINT16        Address;
   UINT8     Channel_Prio;
   UINT8         Type_Cntr;
   //CoE_Cmd
   UINT16        Num_Res_Type;

   //CoE_Specific_Data
   UINT8         ulNodeId;
   UINT16        ulIndex;
   UINT8         ulSubIndex;
   UINT32        ulDataCnt;
};

struct CAN_Packet_Data
{
   UINT8        abSdoData[100];
};
struct CAN_Packet
{
   CAN_Packet_Header          can_pkt_header;
   CAN_Packet_Body            can_pkt_body;
   CAN_Packet_Data            can_pkt_data;
};

struct SDO_Services_Packet
{
   EthernetHeader              ethernet_header;
   EthercatFrameHeader         ethercat_frame_header;
   EthercatDatagramHeader
   ethercat_datagram_header;
   CAN_Packet                  can_pkt;
   WorkingCounter              wkc;
};
```

Algorithm 2: CoE frame structure (header file: ecat_packet.h).

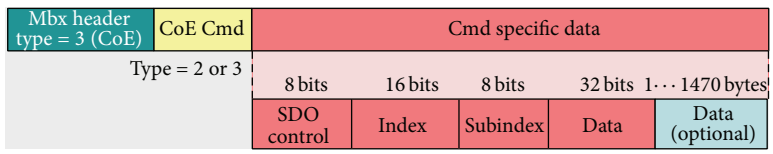


Figure 17: CANopen over EtherCAT devicearchitecture (slide 131 of [7]).

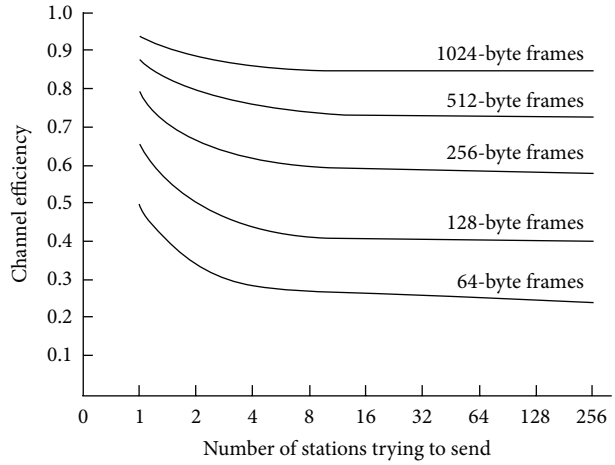


Figure 18: Channel efficiency over the total packet size with TCP overhead 20 bytes [25].

deterministic. Application software must detect this and retry the transmission.

UDP/Ethernet and EtherCAT share the same physical layer. Cabling and switching infrastructure are identical. Acceptable, though still nondeterministic, performance over Ethernet requires bandwidth utilization of under about 30%. By contrast, EtherCAT can utilize over 90% of the media bandwidth.

A JAUS Subsystem can be large and complex, and complexity can increase over time as things are added. Things could be added without the oversight of a system engineer.

Ethernet is susceptible to a DOS (Denial of Service) attack. Nodes are computers; they have operating systems which could be infected by malware, or an infected computer could be plugged into the network. Flooding a network with traffic will effectively shut it down. This is particularly easy to do with UDP, since, at the protocol level, UDP has no congestion control (unlike TCP). An EtherCAT network should be more robust, since devices enforce congestion control, and there would be no temptation to plug a computer into the network.

8. Conclusions

In this research we have created a basic JAUS implementation in order to evaluate the performance of the standard at its core. Even in its simplest form, JAUS is unable to perform at the level of CAN for a few different reasons. The hierarchical structure through which messages must be passed contributes significantly to latency issues. Additionally, the UDP protocol provides no means for prioritizing messages, making it virtually impossible to make reasonable guarantees of message delivery times, and is simply slower than CAN with higher bus speeds. Consequently, JAUS/UDP is not a suitable communication standard for internal control systems of UVs.

Although JAUS is not suitable for communication in internal, real-time systems in UVs, it can still be used for the purpose of interoperability. By using EtherCAT for internal controls and JAUS as the subsystem level protocol, both interoperability and performance can be achieved, as we presented in the paper. Manufacturers are still able to use their existing internal systems and need only provide a way to interface, or translate between, JAUS and their proprietary methods.

Regarding our JEBridge solution, we note that closed loop control systems require deterministic time, which JAUS does not guarantee. Thus, we do not recommend Level 3 JAUS Compliance and recommend using EtherCAT in its place. EtherCAT does not compromise the goal of "reusable components" [19–21], because of the range of available EtherCAT modules [26]. In addition, there are advantages to forgoing JAUS Level 2 Compliance and using EtherCAT even when critical timing is not a requirement, since EtherCAT provides a robust backbone communication mechanism. Of course, we remain solid supporters of JAUS Level 1 Compliance.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by Dual Use Projects of MKE (Ministry of Knowledge and Economy) contract titled "Development of Quadruped Legged Robot Platform Technology." Although JAUS was originally an initiative starting in 1998 by the United States Department of Defense to develop an open architecture for the domain of unmanned systems, it is now an open international standard that is supervised by SAE International and the research group planned to develop a robot architecture based upon it. An earlier version of this paper was approved by ADD Dual Use Technology Center for Public Release. The authors also would like to thank anonymous reviewers for their valuable comments and suggestions which lead to substantial improvements of their paper.

References

[1] United States Senate, "Fiscal Year 2001 Defense Authorization Bill," Sec 217.

[2] SAE, "Class C application requirement considerations-SAE J2056/1," in *SAE Handbook*, pp. 23.366–23.371, 1993.

[3] S. Rowe and C. Wagner, "An Introduction to the Joint Architecture for Unmanned Systems (JAUS), Open Skies," 2008, http://www.openskies.net/papers/07F-SIW-089%20Introduction%20to%20JAUS.pdf.

[4] S. J. Lee, D. M. Lee, and J. C. Lee, "Development of communication framework for unmanned ground vehicle," in *Proceedngs of the International Conference on Control, Automation and Systems (ICCAS '08)*, pp. 604–607, Seoul, Republic of Korea, October 2008.

[5] D. Barber, S. Leontyev, B. Sun, L. Davis, D. Nicholson, and J. Y. C. Chen, "The mixed-initiative experimental testbed for collaborative human robot interactions," in *Proceedings of the International Symposium on Collaborative Technologies and Systems (CTS '08)*, pp. 483–489, Irvine, Calif, USA, May 2008.

[6] M. Sugiura, K. Kobayashi, K. Watanabe, and T. Ohkubo, "Development of unmanned ground vehicle for IGVC JAUS challenge," in *Proceedings of the SICE Annual Conference*, pp. 2719–2722, Tokyo, Japan, August 2008.

[7] EtherCAT Technology Group, "EtherCAT Communication," 2008, http://lectoraatmechatronica.wikispaces.com/file/view/EtherCAT%20communication.pdf/346392478/EtherCAT%20communication.pdf.

[8] N. Navet, Y. Song, F. Simonot-Lion, and C. Wilwert, "Trends in automotive communication systems," *Proceedings of the IEEE*, vol. 93, no. 6, pp. 1204–1223, 2005.

[9] K. Tindell and A. Burns, "Guaranteeing message latencies on Controller Area Network (CAN)," in *Proceedings of the 1st International CAN Conference*, pp. 1–11, Mainz, Germany, 1994.

[10] H. Woo, M. Kim, and J. Kim, "Development of multiple communication using JAUS message set for unmanned system," in *Proceedings of the International Conference on Control, Automation and Systems (ICCAS '07)*, pp. 2374–2377, Seoul, Republic of Korea, October 2007.

[11] N. C. Audsley, A. Burns, M. F. Richardson, and K. Tindell, "Applying new scheduling theory to static priority pre-emptive scheduling," *Software Engineering Journal*, vol. 8, no. 5, pp. 284–2292, 1993.

[12] J. Albus, H. Huang, E. R. Messina et al., "4D/RCS: a reference model architecture for unmanned vehicle systems version 2.0," 2002, http://www.robotictechnologyinc.com/images/upload/file/4DRCS.pdf.

[13] "EtherCAT Master, Cross Platform Stack," 2011, http://www.esd-electronics-usa.com/Shared/Handbooks/EtherCATMasterDevelopersManual.pdf.

[14] CAN in Automation (CiA), http://www.can-cia.org/.

[15] S. Talbot and S. Ren, "Comparison of FieldBus systems, CAN, TTCAN, FlexRay and LIN in passenger vehicles," in *Proceedings of the 29th IEEE Conference on Distributed Computing Systems Workshops*, pp. 26–31, Montreal, Canada, 2009.

[16] H. Chen and J. Tian, "Research on the controller area network," in *Proceedings of the International Conference on Networking and Digital Society (ICNDS '09)*, pp. 251–254, Guizhou, China, May 2009.

[17] T. Nolte, H. Hansson, and C. Norstrom, "Probabilistic worst-case response-time analysis for the controller area network," in *Proceedings of the 9th IEEE Real-Time and Embedded Technology and Applications Symposium*, pp. 200–2207, 2003.

[18] R. Davis, A. Burns, R. Bril, and J. Lukkien, "Controller Area Network (CAN) schedulability analysis: refuted, revisited and revised," *Real-Time Systems*, vol. 35, no. 3, pp. 239–272, 2007.

[19] The JAUS, "Reference Architecture Specification," Volume II, Part 1, *Architecture Framework*, Version 3.3, 2007.

[20] The JAUS, "Reference Architecture Specification," Volume II, Part 3, *Message Set*, Version 3.3, 2007.

[21] The JAUS, "Reference Architecture Specification," Volume II, Part 2, *Message Definition*, Version 3.3, 2007.

[22] J. Pedersen, "A practical view and future look at JAUS, RE2," White Paper, May 2006, http://www.thesciencedude.com/projects/RESEARCH/MASSystem/References/JAUSwhitepaper.pdf.

[23] "TinyXML," http://www.grinninglizard.com/tinyxml/.

[24] "CANopen Master, Protocol API," 2009, http://www.hilscher.com/files_manuals/CANopen%20Master%20Protocol%20API.pdf.

[25] A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, Pearson, 5th edition, 2010.

[26] EtherCAT Technology Group, "EtherCAT products," http://www.ethercat.org/en/products.html.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 605749, 6 pages
http://dx.doi.org/10.1155/2013/605749

Research Article

Research on Feature Extraction of Indicator Card Data for Sucker-Rod Pump Working Condition Diagnosis

Yunhua Yu, Haitao Shi, and Lifei Mi

College of Information & Control Engineering, China University of Petroleum (Huadong), Qingdao 266580, China

Correspondence should be addressed to Yunhua Yu; upcyyh1040@gmail.com

Received 8 June 2013; Revised 23 October 2013; Accepted 13 November 2013

Academic Editor: Jason Gu

Three feature extraction methods of sucker-rod pump indicator card data have been studied, simulated, and compared in this paper, which are based on Fourier Descriptors (FD), Geometric Moment Vector (GMV), and Gray Level Matrix Statistics (GLMX), respectively. Numerical experiments show that the Fourier Descriptors algorithm requires less running time and less memory space with possible loss of information due to nonoptimal numbers of Fourier Descriptors, the Geometric Moment Vector algorithm is more time-consuming and requires more memory space, while the Gray Level Matrix Statistics algorithm provides low-dimension feature vectors with more time consumption and more memory space. Furthermore, the characteristic of rotational invariance, both in the Fourier Descriptors algorithm and the Geometric Moment Vector algorithm, may result in improper pattern recognition of indicator card data when used for sucker-rod pump working condition diagnosis.

1. Introduction

The sucker-rod pump system is the most widely used form of artificial lift for the onshore oil well production [1–3]. Approximately, 80% of the oil wells in the world, 90% of those in China, are being produced by the sucker-rod pumps [4, 5]. The maintenance and optimization of a sucker-rod pump system is a costly and time-consuming operation. The indicator card is the relation curve between the load and the displacement of a sucker-rod pump in an intact suck cycle, in which x-axis represents displacement and y-axis represent load [6]. The indicator card is helpful to analyze the down-hole working condition of the sucker-rod pump wells [7], which can judge the operation condition of the sucker-rod pump well and provide reliable proof of high efficiency, reasonable exploitation for the oil well production. While the system is operating, the card can indicate such shape that might be a normal operation or a fault situation. According to different kinds of real-time indicator card data, the pattern recognition and fault diagnosis techniques are used to identify some different curve shapes, locate which kind of abnormal situation is, and interpret why the fault occurs [8]. Therefore, the correct and quick identification of the sucker-rod pump indicator card is essential to the fault diagnosis of down-hole working condition. The automatic fault diagnosis of sucker-rod pump working condition is a visual interpretation process [9]. Nowadays, the traditional methods of interpretation are not suitable for the automatic fault diagnosis of the down-hole conditions. And several signal processing methods, such as artificial neural network (ANN) [10] and fuzzy support vector machine (FSVM) [11], have been studied and applied to pattern recognition of indicator cards to improve the accuracy and efficiency of sucker-rod pump system fault diagnosis. Because there are more fault patterns and less fault samples in the sucker-pod pump working conditions, the above approaches have their limits, respectively. In recent years, a method called biomimetic pattern recognition (BPR) [12] has been proposed, which is based on the principle of homology continuity and is suitable to recognize and classify those objects with more pattern types and less samples.

This paper is focused to study, compare, and select the proper feature extraction methods used to analyze the sucker-rod pump indicator card data before pattern classification. The optimal feature extraction algorithm of sucker-rod indicator card data followed by the matching pattern recognition

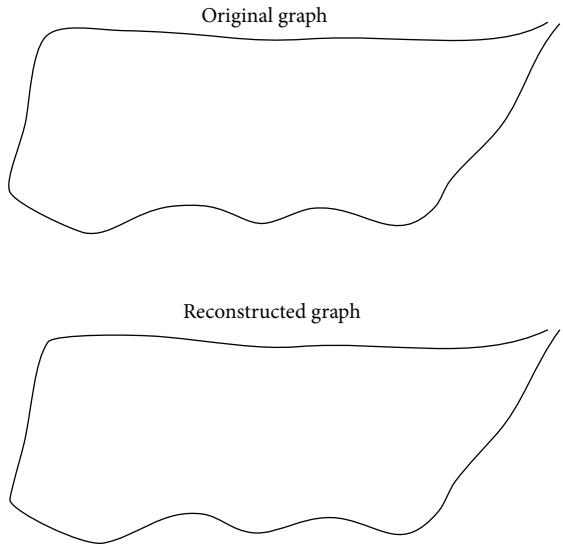


Figure 1: Result of indicator card feature extraction.

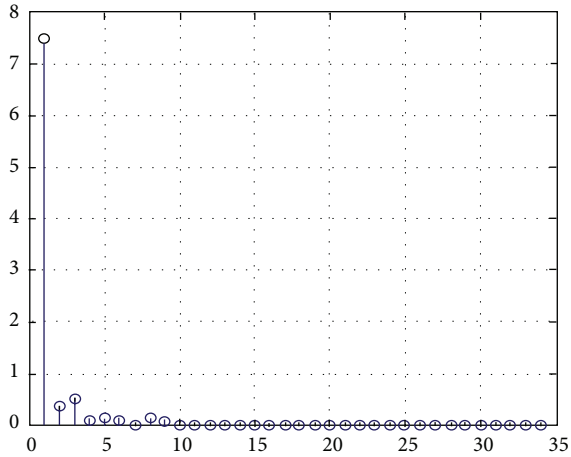


Figure 2: Amplitude spectrum of indicator card.

is helpful to locate the exact fault type of working sucker-rod pump, which is of great significance in improving crude oil production as well as preventing some possible safety accidents. The different feature extraction algorithms of indicator card are based on Fourier Descriptors, Geometric Moment Vector, Gray Level Matrix Statistics, Area and Difference Curve, respectively [13].

The area-based method can only determine some limited fault types of narrow-band distribution indicator card, such as pump spraying fault [14]. The difference curve-based method could not determine some greatly dangerous pump failures, such as stuck pump fault [15]. In this paper, we focus on three feature extraction algorithms of the indicator card, which are based on Fourier Descriptors (FD), Geometric Moment Vector (GMV), and Gray Level Matrix Statistics (GLMS), respectively. With numerical experimental simulation and analysis, three different algorithms are compared in terms of the consumption of time and the complexity of memory space.

2. Feature Extraction Methods of Indicator Card Data

2.1. Method I: Algorithm Based on Fourier Descriptors. Fourier transformation is a popular method for reconstruction and classification of image. It generates a complete set of complex numbers-the Fourier Descriptors (FD), which represent the object shape in a frequency domain [16].

To reduce the computational complexity, the polygonal approximation method is used to cross out those redundant indicator card points (data). The procedures are as follows.

Firstly, according to a given value, which is called D, we traverse all the digital pixels (data) of the indicator card curve in order to choose the feature pixels (data) of the polygon, which meet the condition with maximum curvature of a certain length curve.

Secondly, we store those feature data to an array.

The given value (D) is assigned as 0.008 according to comparison of different computation procedures. Take the pump-on-touch fault as an example; 34 feature pixels (data) are extracted from the original 702 pixels. Figure 1 shows the reconstructed graphic curve of the pump-on-touch fault compared with the original one.

It is reasonable to employ Discrete Fourier Transforms (DFT) to obtain FD [17]. However, some sample errors may be produced during the sampling process.

After the feature extraction of polygonal approximation, the Fourier Transform (FT) of each polyline is employed to avoid the possible error caused by sampling and to improve the speed and accuracy of calculation.

The Fourier Descriptors, denoted by $Z(k)$, are generated by such math formula as follows [18]:

$$p(l) = x(l) + jy(l) = \sum_{k=-\infty}^{+\infty} z(k)\, e^{j(2\pi kl/L)},$$

$$z(k) = \frac{1}{L}\int_0^L p(l)\, e^{-j(2\pi kl/L)}\, dl \tag{1}$$

$$= \frac{1}{L}\sum_{n=0}^{N-1}\int_{l_n}^{l_{n+1}} p(l)\, e^{-j(2\pi kl/L)}\, dl,$$

where L denotes the polygons perimeter. (x_n, y_n) denotes the coordinates of vertex $\mathbf{P}_n$. l_n denotes the accumulated length sum of those short polylines between $\mathbf{P}_0$ and $\mathbf{P}_n$.

The amplitude spectrum of the indicator card obtained after Discrete Fourier Transforms (DFT) is shown in Figure 2. The lower frequency elements of Fourier Descriptors contain the most important information of the indicator card while the higher frequency ones contain less information. Therefore, a subset of the Fourier Descriptors can be used to discriminate different shapes of curves. In this paper, k is set as 10, which means the first 10 FDs are used.

The Fourier Descriptors are affected by the location of vertex $\mathbf{P}_n$, the scale and direction of curve. To eliminate these effects, the normalization is employed. If the given curve object is firstly magnified by r times, secondly its starting position is shifted by a, and then it is rotated by φ degree, finally it is translated by the displacement (x_0, y_0). We get a

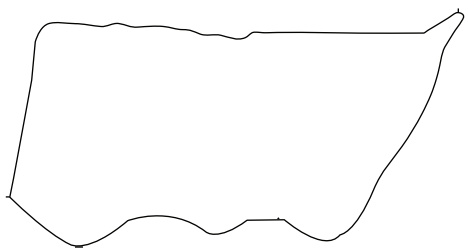


Figure 3: Image Binarization and refinement.

new curve object, and the new Fourier Transform coefficient of the new curve is

$$\begin{aligned} z'(k) &= F\left[\left(x' + iy'\right) re^{j\varphi} + \left(x_0 + iy_0\right)\right] \\ &= re^{j\varphi} F\left(x' + iy'\right) + F\left(x_0 + iy_0\right) \\ &\Longrightarrow \frac{r\left\| e^{j\varphi} e^{j(2\pi/L)ka} z(k)\right\|}{r\left\| e^{j\varphi} e^{j(2\pi/L)ka} z(1)\right\|} = \frac{\|z(k)\|}{\|z(1)\|}. \end{aligned} \tag{2}$$

The effect of modulus and phase changes on Fourier Descriptors is eliminated due to the identically equal ratio as shown in the formula (2).

2.2. Method II: Algorithm Based on Geometric Moment Vector. In image processing field, Geometric Moment Vector (GMV) can be used as an important feature to represent objects due to its translation invariance, rotation invariance, and scale invariance.

Since the scanned image of indicator card is grayscale and the image outline is not smooth, it is necessary to preprocess the image by using binarization and refinement technology. Figure 3 is a single-pixel binary image after the binarization and refinement processing.

The seven two-dimensional moment invariants are as follows, any of which is not sensitive for translation, scaling, mirroring, and rotation [18]:

$$\begin{aligned} f_1 &= \eta_{20} + \eta_{02}, \\ f_2 &= \left(\eta_{20} - \eta_{02}\right)^2 + 4\eta_{11}^2, \\ f_3 &= \left(\eta_{30} - 3\eta_{12}\right)^2 + \left(3\eta_{21} - \eta_{03}\right)^2, \\ f_4 &= \left(\eta_{30} + \eta_{12}\right)^2 + \left(\eta_{21} + \eta_{03}\right)^2, \\ f_5 &= \left(\eta_{30} - 3\eta_{12}\right)\left(\eta_{30} + \eta_{12}\right) \\ &\quad \times \left[\left(\eta_{30} + \eta_{12}\right)^2 - 3\left(\eta_{21} + \eta_{03}\right)^2\right] \\ &\quad + \left(3\eta_{21} - \eta_{03}\right)\left(\eta_{21} + \eta_{03}\right) \\ &\quad \times \left[3\left(\eta_{30} + \eta_{12}\right)^2 - \left(\eta_{21} + \eta_{03}\right)^2\right], \\ f_6 &= \left(\eta_{20} - \eta_{02}\right)\left[\left(\eta_{30} + \eta_{12}\right)^2 - \left(\eta_{21} + \eta_{03}\right)^2\right] \\ &\quad + 4\eta_{11}\left(\eta_{30} + \eta_{12}\right)\left(\eta_{21} + \eta_{03}\right), \\ f_7 &= \left(3\eta_{21} - \eta_{03}\right)\left(\eta_{30} + \eta_{12}\right) \\ &\quad \times \left[\left(\eta_{30} + \eta_{12}\right)^2 - 3\left(\eta_{21} + \eta_{03}\right)^2\right] \\ &\quad + \left(3\eta_{12} - \eta_{30}\right)\left(\eta_{21} + \eta_{03}\right) \\ &\quad \times \left[3\left(\eta_{30} + \eta_{12}\right)^2 - \left(\eta_{21} + \eta_{03}\right)^2\right]. \end{aligned} \tag{3}$$

2.3. Method III: Algorithm Based on Gray Level Matrix Statistics. The grid method is one of the traditional image feature extraction methods. The processing steps of the grid method are as follows.

Firstly, we divide the image of indicator card into a number of small grids with same size and shape, in the horizontal and vertical direction, respectively. Then, we mark the grids which are traversed by the curve of indicator card. Finally, we can obtain the feature parameters of the indicator card image.

In this paper, we use a Gray Level Matrix Statistics (GLMS) feature extraction method which is based on grid method.

Before the GLMS feature extraction, the indicator card curve should be converted to grayscale graphic matrix. The steps of GLMS feature extraction algorithm are as follows.

(1) The mesh of grayscale matrix is initialized: if a grid is traversed by the indicator card curve, then the gray value is assigned as "1".

(2) According to the gray contour principle, other grids are assigned as such gray value: if the grid is located at the inside of the curve, then the gray value is equal to the initial value plus n for its n grids distance away from the curve; if the grid is located at the outer region of the curve, then the gray value is equal to the initial value minus n for its n grids distance away from the curve.

(3) Finally, we can get 6 statistic parameters of the gray level matrix [19].

3. Numerical Experiments and Results

Three typical fault indicator cards are shown in Figures 4, 5, and 6.

According to the Method I (Fourier Descriptors), we compute 12 normalized subsets of the Fourier Descriptors (FDs) for 12 different typical fault of indicator cards. Each set includes 10 main FDs, which is called a feature vector denoted as $F = [\mathrm{FD}_1, \mathrm{FD}_2, \mathrm{FD}_3, \mathrm{FD}_4, \mathrm{FD}_5, \mathrm{FD}_6, \mathrm{FD}_7, \mathrm{FD}_8, \mathrm{FD}_9, \mathrm{FD}_{10}]$. Table 1 shows the numerical result.

According to the Method II (Geometric Moment Vector), we compute 12 subsets of the Geometric Moment Vectors (GMV) for 12 different typical fault of indicator cards. Each set includes 7 GMVs, which is called a feature vector denoted as $F = [f_1, f_2, f_3, f_4, f_5, f_6, f_7]$. Table 2 shows the numerical result.

Table 1: Normalized Fourier Descriptors of 12 typical fault indicator cards.

	FD_1	FD_2	FD_3	FD_4	FD_5	FD_6	FD_7	FD_8	FD_9	FD_{10}
Top pump bumping	1	0.04825	0.06673	0.00921	0.01938	0.01228	0.01789	0.00373	0.00615	0.00368
Bottom pump bumping	1	0.04190	0.06150	0.01079	0.02105	0.00580	0.02075	0.00018	0.00525	0.00382
Broken-dropped of sucker rod	1	0.00256	0.07137	0.04122	0.00181	0.00423	0.01041	0.00278	0.00612	0.00312
Insufficient fluid supply	1	0.09135	0.07921	0.03970	0.02756	0.00268	0.00274	0.02133	0.00656	0.00519
Leakage of traveling valve	1	0.02729	0.08138	0.01280	0.02071	0.02071	0.01073	0.00803	0.00936	0.00458
Leakage of standing valve	1	0.04080	0.08172	0.01214	0.02129	0.01990	0.00740	0.00619	0.00890	0.00371
Leakage of both valves	1	0.00256	0.08158	0.01256	0.02517	0.01928	0.00683	0.00725	0.00912	0.00369
Stuck piston	1	0.00823	0.07259	0.01193	0.00158	0.00925	0.00957	0.00458	0.00349	0.00344
Gas lock	1	0.08365	0.06759	0.02358	0.02135	0.00245	0.00256	0.02325	0.00589	0.00623
Liquid hammer	1	0.09195	0.06064	0.02082	0.02073	0.00283	0.00255	0.04195	0.00764	0.00082
Sand inflow	1	0.00925	0.07436	0.00564	0.02577	0.00556	0.01728	0.00058	0.00278	0.00166
Plunger out pump barrel	1	0.04255	0.08212	0.01223	0.02019	0.01625	0.00722	0.00539	0.00693	0.00316

Table 2: Geometric Moment Vectors of typical faults indicator card.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
Top pump bumping	0.4392	0.0658	0.0175	0.0977	0.0038	0.0250	0.0015
Bottom pump bumping	0.4936	0.0928	0.0302	0.1365	0.0084	0.0415	0.0027
Broken-dropped of sucker rod	0.5877	0.1857	0.1261	0.2652	0.0480	0.1142	0.0067
Insufficient fluid supply	0.7061	0.0615	0.1619	0.3834	0.0095	0.0939	0.0951
Leakage of traveling valve	0.4278	0.0646	0.0117	0.0920	0.0030	0.0234	0.0005
Leakage of standing valve	0.4549	0.0827	0.0371	0.1151	0.0073	0.0331	0.0018
Leakage of both valves	0.4997	0.1340	0.0671	0.1636	0.0171	0.0599	0.0016
Stuck piston	3.9352	5.6349	8.3123	49.690	1003.6	117.92	111.89
Gas lock	0.9168	0.1321	0.6305	0.7222	0.1330	0.2230	0.4689
Liquid hammer	0.4799	0.0552	0.0031	0.1165	0.0019	0.0273	0.0011
Sand inflow	0.4677	0.0792	0.0314	0.1161	0.0064	0.0326	0.0028
Plunger out pump barrel	0.4730	0.1149	0.0732	0.1509	0.0156	0.0511	0.0028

Table 3: Gray Level Matrix Statistics of typical faults indicator card.

	f_1	f_2	f_3	f_4	f_5	f_6
Top pump bumping	2.7046	29.6603	−0.3710	2.9569	0.0589	0.9699
Bottom pump bumping	2.0381	29.292	−0.14445	2.4059	0.0568	0.9796
Broken-dropped of sucker rod	0.0078	20.3339	−0.1684	2.4799	0.0710	0.9634
Insufficient fluid supply	−4.1172	59.7812	−0.7084	2.6101	0.0483	0.9753
Leakage of traveling valve	2.4658	40.9686	−0.5122	3.1043	0.0505	0.9742
Leakage of standing valve	3.8604	23.7471	−0.0216	2.2812	0.0618	0.9684
Leakage of both valves	0.3887	35.5648	0.0830	2.4249	0.0518	0.9735
Stuck piston	−9.0381	67.0591	−0.6385	2.2935	0.0454	0.9768
Gas lock	−5.2549	60.6294	−0.6209	2.2908	0.0473	0.9759
Liquid hammer	0.3276	43.7525	−0.6760	3.4674	0.0530	0.9729
Sand inflow	2.6519	25.0238	−0.1072	2.3713	0.0620	0.9682
Plunger out pump barrel	1.3003	37.3400	−0.1966	2.6507	0.0521	0.9734

Table 4: Comparison of three types of algorithms.

	Algorithm based on FDs	Algorithm based on GMV	Algorithm based on GLMS
Running time (second)	0.698414	5.707594	3.292005
Memory space units for each sample	10	7	6

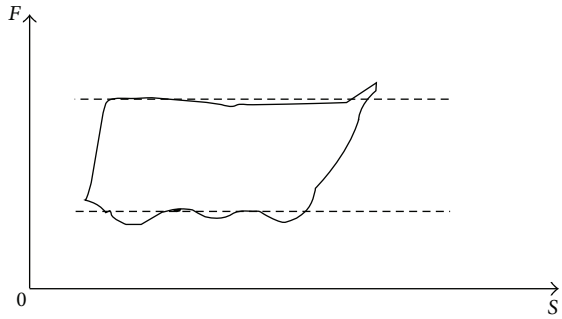


Figure 4: Top Pump bumping.

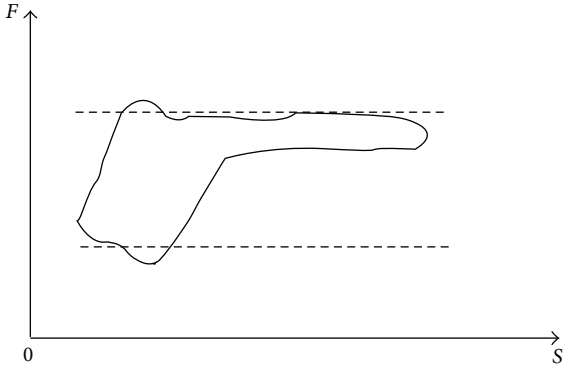


Figure 5: Insufficient fluid supply.

According to the Method III (Gray Level Matrix Statistics), we compute 12 subsets of the Gray Level Matrix Statistic (GLMS) for 12 different typical fault of indicator cards. Each set includes 6 GLMSs, which is called a feature vector denoted as $F = [f_1, f_2, f_3, f_4, f_5, f_6]$. Table 3 shows the numerical result.

4. Conclusion

In this paper, three different feature extraction methods, which are based on Fourier Descriptors, Geometric Moment Vector, and Gray Level Matrix Statistic, respectively, have been analyzed and simulated. The computing speed and memory consuming of these 3 algorithms are compared as shown in Table 4.

Numerical experiments show that the FD algorithm is with high computing speed and more memory space but possible loss of information; because of different numbers of FDs, the GMV algorithm is more time-consuming and less memory consuming, while the GLMS algorithm provides low-dimension vectors with good performance of speed and space.

The characteristic of rotational invariance, both in the FD algorithm and the GMV algorithm, may cause improper pattern recognition of indicator card data when used for sucker-rod pump working condition diagnosis. Further research on feature extraction of indicator card data should continue for better performance.

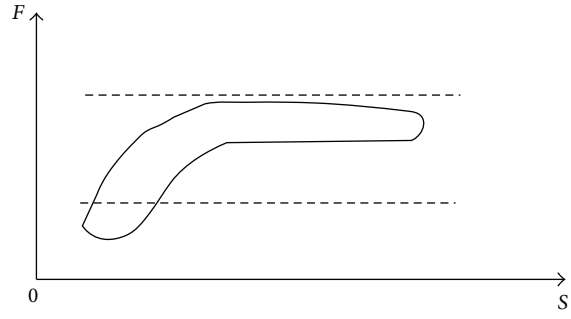


Figure 6: Gas lock.

Acknowledgment

This work is fully supported by Scientific Research Foundation for the Excellent Young Scientist of Shandong Province (no. BX2010DX036).

References

[1] G. P. Schirmer, J. C. P. Toutain, and J. C. Gay, "Use of advanced pattern recognition and knowledge-based system in analyzing dynamometer cards," *SPE Computer Application*, vol. 3, no. 6, pp. 21–24, 1991.

[2] L. Alegre and C. Morooka, "Intelligent diagnosis of rod pumping problems," in *Proceedings of the 68th Annual Technical Conference*, vol. 3, pp. 97–108, SPE, 1993.

[3] G. Han and X. Wu, "Application of dynamometer card identification in diagnosis of working-condition for suck rod pump," *Oil Drilling & Production Technology*, vol. 25, no. 5, pp. 70–74, 2003.

[4] H. Tripp, "A review: analyzing beam-pumped wells," *Journal of Petroleum Technology*, vol. 41, no. 5, pp. 457–458, 1989.

[5] G. Nazi and J. Lea, "Application of artificial neural network to pump card diagnosis," *SPE Computer Application*, vol. 6, no. 6, pp. 9–14, 1994.

[6] M. J. Gao, "Portable intelligent instrument for pumping data collection," *Journal of Transducer Technology*, vol. 24, no. 9, pp. 58–62, 2005.

[7] F. Barreto and M. Tygel, "Automatic down-hole card generation and classification," in *Proceedings of the SPE Annual Technical Conference*, pp. 311–318, 1996.

[8] Z. Pan and J. Ge, "An adaptive neural network for identification of dynamometer card," *Acta Petrolei Sinica*, vol. 17, no. 3, pp. 104–109, 1996.

[9] R. Dickinson and J. Jennings, "Use of pattern-recognition techniques in analyzing down-hole dynamometer cards," *SPE Production Engineering*, vol. 5, no. 2, pp. 187–192, 1990.

[10] D. Baodong and L. Mingshan, "Analyzing dynamometer card of well pumping by using artificial nerve network method," *Well Testing*, vol. 7, no. 1, pp. 27–29, 1998.

[11] W. Shi and J. Dai, "FSVM based recognition approach for down-hole pump dynamometer cards," *Oil Field Equipment*, vol. 33, no. 4, pp. 46–48, 2004.

[12] W. Shoujue, "Bionic (topological) pattern recognition-a new model of pattern recognition theory and its applications," *Acta Electronica Sinica*, vol. 30, no. 10, pp. 1417–1420, 2002.

[13] S. C. Yuan, "Simple distinguish way for dynamometer card," *Well Testing*, vol. 14, no. 1, pp. 49–50, 2005.

[14] B. D. Du and M. S. Li, "Analyzing dynamometer card of well pumping by using ANN method," *Well Testing*, vol. 1, no. 7, pp. 27–29, 1998.

[15] S. H. Yang, "Applications of difference curve methods in diagnosis of indicator diagram," *Petroleum Drilling Techniques*, vol. 30, no. 2, pp. 63–64, 2002.

[16] A. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1989.

[17] H. Kauppien and T. Sepanen, "An experiment comparison of auto regressive and Flourier-based descriptors in 2D shape classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 201–207, 1995.

[18] R. C. Gonzalez, *Digital Image Processing*, Electronic Industrial Press, 2005.

[19] W. Wu and G. D. Chen, "Fault diagnosis system for pump work indicating diagram based on neural network and gray-level matrix," *Journal of Xi'an Shiyou University*, vol. 22, no. 3, pp. 119–121, 2007.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 327525, 9 pages
http://dx.doi.org/10.1155/2013/327525

Research Article

Integrity Design for Networked Control Systems with Actuator Failures and Data Packet Dropouts

Xiaomei Qi[1] and Jason Gu[2]

[1] *College of Electrical and Electronic Engineering, Shandong University of Technology, Zibo 255049, China*
[2] *Department of Electrical and Computer Engineering, Dalhousie University, Halifax, NS, Canada B3J 2X4*

Correspondence should be addressed to Jason Gu; jason.gu@dal.ca

Received 8 June 2013; Revised 7 October 2013; Accepted 8 October 2013

Academic Editor: Ying Wang

The integrity design problem of fault tolerant control for networked control system (NCS) with actuator failures and data packet dropouts is investigated. The data packet dropouts in both sensor-controller (S-C) and controller-actuator (C-A) links are described by two switches, which can be modeled as a discrete event system with known rate. After introducing the matrix of actuator failure, the closed-loop NCS is developed, which can be viewed as asynchronous dynamical systems (ADSs). Then, the sufficiency of exponential stability for the NCS is obtained based on the theory of ADSs. The output feedback controllers that can guarantee system stability are also proposed. Finally, two numerical examples are given to demonstrate the validity of our proposed approach.

1. Introduction

Along with the rapid development of communication networks, a great amount of effort has been made on fault-tolerant control (FTC) problems for networked control systems (NCSs) recently. NCSs are distributed systems, which are comprised of controlled plants actuators, sensors, and controllers. The essential feature of NCSs is that the information is exchanged through some form of communication networks (as in [1–5]). The use of a shared network to connect spatially distributed devices results in flexible architectures and generally reduces installation and maintenance costs. Consequently, NCSs have been widely applied to many complex control systems, for example, unmanned aerial vehicles, avionics industries, remote surgery, and rapid transit trains.

However, the insertion of networks also brings some new issues, such as network-induced delay (as in [1, 6–12]) and data packet dropout (as in [1, 13–15]), which make NCSs more vulnerable to faults than conventional systems. As we know, research on FTC strives to make the system stable and retain acceptable performance under the system faults. An important part of FTC is the one specializing in actuator faults, FTC techniques dealing with actuator faults are relevant for practical application and have already been the subject of many publications (as in [16–19]). Therefore, a suitable architecture for FTC of NCSs must take the dynamical behavior of network into consideration (as in [8–12, 14, 15, 20–23]).

A wide range of research has recently been reported dealing with problems related to the FTC for NCSs with network-induced delay (as in [8–11]). As compared to the plentiful works on FTC for NCSs with network-induced delay, only a few attention has been paid to the study of FTC for NCSs with data packet dropout (as in [14, 15, 23]). As we know, packet dropouts can be modeled either as stochastic or deterministic phenomena, such as packet dropouts in both S-C link and C-A link are characterized by Bernoulli process in [14]. Finite-state Markovian process is used to model correlated packet dropouts in [23]. Deterministic models for data dropouts have also been proposed, specified in terms of time averages as in [1, 15], such as packet dropouts in the S-C link modeled by ADSs, discussed in [15].

From the above description, considering data packet dropout in both S-C and C-A links, the problem of FTC for NCSs with actuator failures is still a challenging problem. Thus, this paper is devoted to study FTC for NCSs with actuator failures and packet dropout in both S-C and C-A links. According to the method in [1], data packet dropouts

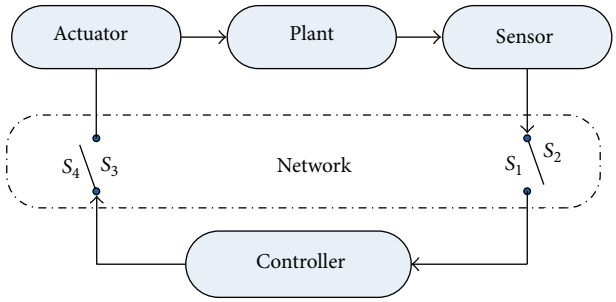


Figure 1: Schematic description of NCSs with random data packet dropouts.

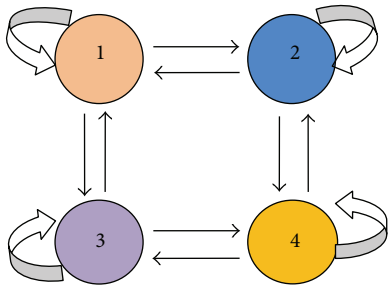


Figure 2: Four-state discrete event system.

in both S-C and C-A links are described by two independent switches, which can be modeled as a discrete event system with known rate. After introducing the matrix of actuator failures, the model of closed-loop NCSs with actuator failures is addressed. It should be mentioned that the closed-loop NCSs can be viewed as ADSs. Then, based on the theory of ADSs, the sufficiency of exponential stability for such NCSs is derived and the output feedback controllers that can guarantee system stability are presented. Finally, two numerical examples are provided to show the effectiveness of the present approach.

The rest of the paper is organized as follows. Section 2 gives the formulation of the problem, and the NCSs with data packet dropout is modeled as a discrete event system. The integrity design of FTC for the NCSs is addressed in Section 3. The output-feedback controllers are also presented in Section 3. Two numerical examples to testify the effectiveness of the proposed method are shown in Section 4. Section 5 makes the concluding remarks.

2. Problem Formulation

The LTI discrete-time system is considered as follows:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k, \end{aligned} \tag{1}$$

where $x_k \in \mathbf{R}^n$, $y_k \in \mathbf{R}^m$, and $u_k \in \mathbf{R}^p$ denote the state vector, the measurable output vector, and the control input vector, respectively. A, B, and C are known matrices with appropriate dimensions.

Networks can be viewed as unreliable data transmission paths, where packet collision and network node failure occasionally occur. When there is a packet collision, instead of repeated retransmission attempts, it might be advantageous to drop the old packet and transmit a new one. Hence, in this study, we focus on the phenomenon of data packet dropout. Figure 1 shows a typical closed loop of NCSs with data packet dropout. Some assumptions in this paper are as follows.

(1) The controller and the actuator are event-driven.

(2) Data are single-packet transmission, and the dropout packet rate is known.

From Figure 2, it can been seen that data packet dropouts in the network can be treated as switches that close at a certain rate. When these switches are closed (position S_1, S_3), data packets containing y_k, u_k are transmitted successfully, whereas when they are open (position S_2, S_4), the outputs of these switches are held at the previous values $\overline{y}_{k-1}$, u_{k-1}, stored in the buff, and the current data packet is missing. Thus, the dynamic model of this network with data packet dropout is given by

$$\begin{aligned} &S_1\colon \overline{y}_k = y_k, \qquad S_2\colon \overline{y}_k = \overline{y}_{k-1}, \\ &S_3\colon u_k = -K_{\theta_k}\overline{y}_k, \qquad S_4\colon u_k = u_{k-1}, \end{aligned} \tag{2}$$

where θ_k is a four-state time-homogenous Markov chain which takes values in index set $\Upsilon = \{1, 2, 3, 4\}$; the rate at which the event $\theta_k = j$ occurs is defined by the following:

$$\begin{aligned} p_{ij} &= \Pr\{\theta_{k+1} = j \mid \theta_k = i\}, \\ \sum_{j=1}^{4} p_{ij} &= 1, \quad \forall i, j \in \Upsilon,\ p_{ij} \geq 0. \end{aligned} \tag{3}$$

From the above discussion, data packet dropouts in the network can be viewed as two independent switches; that is, we can obtain the network model (2) by analyzing the features of these switches. Meanwhile, it is worth noting that the network model (2) can also be a discrete event system, which contains four events as follows.

Event 1: S_1S_3, no data packets are lost in the whole loop.

Event 2: S_2S_3, data packets are lost in the S-C link.

Event 3: S_1S_4, data packets are lost in the C-A link.

Event 4: S_2S_4, data packets's loss exist in both the S-C and the C-A link.

Actually, sensor failures model can be transformed into actuator failures model by changing format of system with faults. So, in this paper, we only discuss actuator failures model. Usually, when a system is under actuator failures, a matrix $L = \mathrm{diag}\{l_1, l_2 \ldots, l_p\}$ will be introduced into this system between its coefficient matrix and the feedback control matrix to describe actuator failures. We have

$$l_j = \begin{cases} 1, & \text{no actuator fault,} \\ 0, & \text{the } j\text{th actuator has fault,} \\ \in (0, 1), & \text{the } j\text{th actuator has partial fault,} \end{cases} \tag{4}$$

where $l_j = 1$ denotes that the jth actuator is in the normal situation, $l_j = 0$ denotes that the jth actuator is in the full fault situation, and $l_j \in (0,1)$ denotes that the jth actuator is in partial fault situation.

Define a new augmented vector $z_k = [x_k^T,\ \overline{y}_{k-1}^T,\ u_{k-1}^T]^T$; then the closed-loop NCS model with actuator failures will be

$$z_{k+1} = \widetilde{\Phi}_{\theta_k} z_k, \tag{5}$$

where

$$\widetilde{\Phi}_1 = \begin{bmatrix} A - BLK_1C & 0 & 0 \\ C & 0 & 0 \\ -LK_1C & 0 & 0 \end{bmatrix}, \quad \widetilde{\Phi}_2 = \begin{bmatrix} A & 0 & B \\ C & 0 & 0 \\ 0 & 0 & I \end{bmatrix},$$
$$\widetilde{\Phi}_3 = \begin{bmatrix} A & -BLK_3 & 0 \\ 0 & I & 0 \\ 0 & -LK_3 & 0 \end{bmatrix}, \quad \widetilde{\Phi}_4 = \begin{bmatrix} A & 0 & B \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix}. \tag{6}$$

Up till now, the state space expressions for different discrete events of the closed-loop NCS are acquired.

3. Integrity Design of FTC for the NCS

In this section, the integrity design of FTC for the NCS (5) is discussed. As we know, the integrity design is that system can keep asymptotically stable under some sensors failures or actuators failures and is one of the important means for the FTC. Meanwhile, because system (5) can also be viewed as ADSs, which are systems that incorporate continuous and discrete dynamics, the continuous dynamics are governed by differential or difference equations, whereas the discrete dynamics are governed by finite automata that are driven asynchronously by external discrete events with fixed rates [24, 25]. Then before studying the problem of integrity design, an important lemma about ADSs should be introduced.

Lemma 1 (as in [24]). *Given an ADS as follows:*

$$x(k+1) = f_s(x(k)), \quad s = 1, 2, \ldots, N. \tag{7}$$

Suppose a Lyapunov function $V : R_n \to R_+$ and

$$\beta_1 \|x\|^2 \le V(x) \le \beta_2 \|x\|^2, \tag{8}$$

where $\beta_{1,2} > 0$. The ADS (7) is exponentially stable with the decay rate δ if there exist the scalars $\alpha_1, \alpha_2, \ldots, \alpha_M > 0$, satisfying

$$\alpha_1^{r_1}\alpha_2^{r_2}\cdots\alpha_M^{r_M} > \alpha > 1,$$
$$V(x(k+1)) - V(x(k)) \le \left(\alpha_{s_1}^{-2}\alpha_{s_2}^{-2}\cdots\alpha_{s_{M_s}}^{-2} - 1\right) V(x(k)), \tag{9}$$

where M_s is the number of events to each discrete state, M is the number of events in total, and $r_{1,\ldots,M}$ are corresponding to each event rate.

Remark 2. Note that the $\alpha_{s_j}, j = 1, \ldots, M_s$ belongs to the set $\{\alpha_1, \alpha_2, \ldots, \alpha_M\}$. And Lemma 1 implies that the Lyapunov function $V(x)$ does not have to decrease monotonically at some rate δ along all trajectories, but rather it should decrease at a rate δ on the average.

Theorem 3. *If r_1, r_2 represents the rate of events S_1, S_3, respectively and $\widehat{r}_1, \widehat{r}_2, \widehat{r}_3, \widehat{r}_4$ are the rates of events S_1S_3, S_1S_4, S_2S_3, S_2S_4 in NCS (5), then $\widehat{r}_1, \widehat{r}_2, \widehat{r}_3, \widehat{r}_4$ can be described as*

$$\widehat{r}_1 = r_1 r_2, \qquad \widehat{r}_2 = r_1(1 - r_2),$$
$$\widehat{r}_3 = (1 - r_1) r_2, \qquad \widehat{r}_4 = (1 - r_1)(1 - r_2). \tag{10}$$

Proof. Since the switches in Figure 1 are independent, then S_1, S_2, S_3, S_4 are also independent events. According to the probability theory, the probabilities of these four independent events can be obtained as follows:

$$\Pr(S_1S_3) = \Pr(S_1)\Pr(S_3) = r_1 r_2,$$
$$\Pr(S_1S_4) = \Pr(S_1)\Pr(S_4) = r_1(1 - r_2),$$
$$\Pr(S_2S_3) = \Pr(S_2)\Pr(S_3) = (1 - r_1) r_2, \tag{11}$$
$$\Pr(S_2S_4) = \Pr(S_2)\Pr(S_4) = (1 - r_1)(1 - r_2).$$

Then, Theorem 3 can be proved. □

Theorem 4. *The NCS (5) is exponentially stable with the decay rate δ if there exist $\delta > 0$, $\alpha_i > 0$, $i = 1, 2, 3, 4$, positive definite matrices P, Q, R, and matrices K_1, K_3, satisfying the following:*

$$\alpha_1^{r_1 r_2}\alpha_2^{r_1(1-r_2)}\alpha_3^{(1-r_1)r_2}\alpha_4^{(1-r_1)(1-r_2)} > \delta > 1, \tag{12}$$

$$\begin{bmatrix} -\alpha_1^{-2}P^{-1} & 0 & 0 & P^{-1}A^T - P^{-1}C^TK_1^TL^TB^T & P^{-1}C^T & P^{-1}C^TK_1^TL^T \\ 0 & -\alpha_1^{-2}Q^{-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & -\alpha_1^{-2}R^{-1} & 0 & 0 & 0 \\ AP^{-1} - BLK_1CP^{-1} & 0 & 0 & -P^{-1} & 0 & 0 \\ CP^{-1} & 0 & 0 & 0 & -Q^{-1} & 0 \\ LK_1CP^{-1} & 0 & 0 & 0 & 0 & -R^{-1} \end{bmatrix} < 0, \tag{13}$$

$$\begin{bmatrix} -\alpha_2^{-2}P^{-1} & 0 & 0 & P^{-1}A^T & P^{-1}C^T & 0 \\ 0 & -\alpha_2^{-2}Q^{-1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \left(1-\alpha_2^{-2}\right)R^{-1} & R^{-1}B^T & 0 & R^{-1} \\ AP^{-1} & 0 & BR^{-1} & -P^{-1} & 0 & 0 \\ CP^{-1} & 0 & 0 & 0 & -Q^{-1} & 0 \\ 0 & 0 & R^{-1} & 0 & 0 & -R^{-1} \end{bmatrix} < 0, \tag{14}$$

$$\begin{bmatrix} -\alpha_3^{-2}P^{-1} & 0 & 0 & P^{-1}A^T & 0 & 0 \\ 0 & \left(1-\alpha_3^{-2}\right)Q^{-1} & 0 & -Q^{-1}K_3^TL^TB^T & Q^{-1} & -Q^{-1}K_3^TL^T \\ 0 & 0 & \alpha_3^{-2}R^{-1} & 0 & 0 & 0 \\ AP^{-1} & -BLK_3Q^{-1} & 0 & -P^{-1} & 0 & 0 \\ 0 & Q^{-1} & 0 & 0 & -Q^{-1} & 0 \\ 0 & -LK_3Q^{-1} & 0 & 0 & 0 & -R^{-1} \end{bmatrix} < 0, \tag{15}$$

$$\begin{bmatrix} -\alpha_4^{-2}P^{-1} & 0 & 0 & P^{-1}A^T & 0 & 0 \\ 0 & \left(1-\alpha_4^{-2}\right)Q^{-1} & 0 & 0 & Q^{-1} & 0 \\ 0 & 0 & \left(1-\alpha_4^{-2}\right)R^{-1} & R^{-1}B^T & 0 & R^{-1} \\ AP^{-1} & 0 & BR^{-1} & -P^{-1} & 0 & 0 \\ 0 & Q^{-1} & 0 & 0 & -Q^{-1} & 0 \\ 0 & 0 & R^{-1} & 0 & 0 & -R^{-1} \end{bmatrix} < 0. \tag{16}$$

Proof. From Lemma 1 and (10), we can get (12) directly.

Next, for nonzero $x_k, \overline{y}_{k-1}, u_{k-1}$, choose the Lyapunov function as follows:

$$V = x_k^T P x_k + \overline{y}_{k-1}^T Q \overline{y}_{k-1} + u_{k-1}^T R u_{k-1}. \tag{17}$$

For simplicity, let $x_k = x$, $\overline{y}_{k-1} = \overline{y}$, and $u_{k-1} = \widehat{u}$. Then, each discrete event of NCS will be discussed, respectively.

Event 1. If event $S_1S_3 : \overline{y}_k = y_k$, $u_k = -LK_1\overline{y}_k$ occurs, it follows that

$$\begin{aligned} &V(z_{k+1}) - \alpha_1^{-2}V(z_k) \\ &\quad = x^TA^TPAx - x^TA^TPBLK_1Cx \\ &\qquad - x^TC^TK_1^TL^TB^TPAx + x^TC^TQCx \\ &\qquad + x^TC^TK_1^TL^TB^TPBLK_1Cx \\ &\qquad + x^TC^TK_1^TL^TRLK_1Cx \\ &\qquad - \alpha_1^{-2}x^TPx - \alpha_1^{-2}\overline{y}^TQ\overline{y} - \alpha_1^{-2}\widehat{u}^TR\widehat{u}. \end{aligned} \tag{18}$$

So, we have

$$V(z_{k+1}) - \alpha_1^{-2}V(z_k) = \begin{bmatrix} x^T & \overline{y}^T & \widehat{u}^T \end{bmatrix} H_1 \begin{bmatrix} x \\ \overline{y} \\ \widehat{u} \end{bmatrix}, \tag{19}$$

where

$$H_1 = \begin{bmatrix} \Pi_{11} & 0 & 0 \\ 0 & -\alpha_1^{-2}Q & 0 \\ 0 & 0 & -\alpha_1^{-2}R \end{bmatrix},$$

$$\begin{aligned} \Pi_{11} = {} & A^TPA + A^TC^TQCA \\ & - C^TK_1^TL^TB^TPA - A^TPBLK_1C \\ & + C^TK_1^TL^TB^TPBLK_1C \\ & + C^TK_1^TL^TRLK_1C - \alpha_1^{-2}P. \end{aligned} \tag{20}$$

By Schur complement, $H_1 < 0$ is equivalent to

$$\begin{bmatrix} -\alpha_1^{-2}P & 0 & 0 \\ 0 & -\alpha_1^{-2}Q & 0 \\ 0 & 0 & -\alpha_1^{-2}R \end{bmatrix} + \begin{bmatrix} A^T - C^TK_1^TL^TB^T & C^T & C^TK_1^TL^T \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} -P^{-1} & 0 & 0 \\ 0 & -Q^{-1} & 0 \\ 0 & 0 & -R^{-1} \end{bmatrix}^{-1} \begin{bmatrix} A - BLK_1C & 0 & 0 \\ C & 0 & 0 \\ LK_1C & 0 & 0 \end{bmatrix} < 0. \tag{21}$$

Then,

$$\begin{bmatrix} -\alpha_1^{-2}P & 0 & 0 & A^T - C^T K_1^T L^T B^T & C^T & C^T K_1^T L^T \\ 0 & -\alpha_1^{-2}Q & 0 & 0 & 0 & 0 \\ 0 & 0 & -\alpha_1^{-2}R & 0 & 0 & 0 \\ A - BLK_1C & 0 & 0 & -P^{-1} & 0 & 0 \\ C & 0 & 0 & 0 & -Q^{-1} & 0 \\ LK_1C & 0 & 0 & 0 & 0 & -R^{-1} \end{bmatrix} < 0. \tag{22}$$

Next, after pre- and postmultiplying inequality (22) by $\mathrm{diag}\{P^{-1}, Q^{-1}, R^{-1}, I, I, I\}$, we have (13). Therefore, the following inequality holds:

$$V(z_{k+1}) - \alpha_1^{-2} V(z_k) < 0. \tag{23}$$

Then, according to the Lemma 1, the NCS (5) in discrete events $\theta_k = 1$ is exponentially stable.

Event 2. If event S_2S_3 : $\overline{y}_k = \overline{y}_{k-1}$, $u_k = -LK_3\overline{y}_k$ occurs, then we get

$$\begin{aligned} &V(x(k+1)) - \alpha_3^{-2} V(x(k)) \\ &\quad = x^T A^T PAx - x^T A^T PBLK_3\overline{y} \\ &\qquad - \overline{y}^T K_3^T L^T B^T PAx + \overline{y}^T K_3^T L^T B^T PBLK_3\overline{y} \\ &\qquad + \overline{y}^T Q\overline{y} + \overline{y}^T K_3^T L^T RLK_3\overline{y} \\ &\qquad - \alpha_3^{-2} x^T Px - \alpha_3^{-2}\overline{y}^T Q\overline{y} - \alpha_3^{-2}\widehat{u}^T R\widehat{u}. \end{aligned} \tag{24}$$

So, we have

$$V(z_{k+1}) - \alpha_3^{-2} V(z_k) = \begin{bmatrix} x^T & \overline{y}^T & \widehat{u}^T \end{bmatrix} H_3 \begin{bmatrix} x \\ \overline{y} \\ \widehat{u} \end{bmatrix}, \tag{25}$$

where

$$H_3 = \begin{bmatrix} A^T PA - \alpha_3^{-2}P & -A^T PBLK_3 & 0 \\ -K_3^T L^T B^T PA & \Pi_{33} & 0 \\ 0 & 0 & 0 \end{bmatrix}, \tag{26}$$

where $\Pi_{33} = K_3^T L^T B^T PBLK_3 + Q + K_3^T L^T RLK_3 - \alpha_2^{-2}Q$.

Applying Schur complement theory, $H_3 < 0$ is equivalent to

$$\begin{bmatrix} -\alpha_3^{-2}P & 0 & 0 & A^T & 0 & 0 \\ 0 & \left(1 - \alpha_3^{-2}\right)Q & 0 & -K_3^T L^T B^T & I & -K_3^T L^T R \\ 0 & 0 & \alpha_2^{-2}R & 0 & 0 & 0 \\ A & -BLK_3 & 0 & -P^{-1} & 0 & 0 \\ 0 & I & 0 & 0 & -Q^{-1} & 0 \\ 0 & -LK_3 & 0 & 0 & 0 & -R^{-1} \end{bmatrix} < 0. \tag{27}$$

Next, after pre- and postmultiplying inequality (27) by $\mathrm{diag}\{P^{-1}, Q^{-1}, R^{-1}, I, I, I\}$, we have (15). Therefore, the inequality $V(z_{k+1}) - \alpha_3^{-2}V(z_k) < 0$ holds.

If event 3 and event 4 occur, that is, S_1S_4: $\overline{y}_k = y_k$, $u_k = u_{k-1}$ and S_2S_4: $\overline{y}(k) = \overline{y}_{k-1}$, $u_k = u_{k-1}$, then we get

$$\begin{aligned} &V(z(k+1)) - \alpha_2^{-2} V(z(k)) \\ &\quad = x^T A^T PAx + x^T A^T PB\widehat{u} + \widehat{u}^T B^T PAx + \widehat{u}^T B^T PB\widehat{u} \\ &\qquad + \widehat{u}^T R\widehat{u} + x^T C^T QCx - \alpha_2^{-2} x^T Px - \alpha_2^{-2}\overline{y}^T Q\overline{y} \\ &\qquad - \alpha_2^{-2}\widehat{u}^T R\widehat{u}, \end{aligned}$$

$$\begin{aligned} &V(x(k+1)) - \alpha_4^{-2} V(x(k)) \\ &\quad = x^T A^T PAx + x^T A^T PB\widehat{u} + \widehat{u}^T B^T PAx + \widehat{u}^T B^T PB\widehat{u} \\ &\qquad + \widehat{u}^T R\widehat{u} + \overline{y}^T Q\overline{y} - \alpha_4^{-2} x^T Px - \alpha_4^{-2}\widehat{u}^T R\widehat{u} - \alpha_4^{-2}\overline{y}^T Q\overline{y}, \end{aligned}$$

$$H_2 = \begin{bmatrix} A^T PA + C^T QC - \alpha_2^{-2}P & 0 & A^T PB \\ 0 & -\alpha_2^{-2}Q & 0 \\ B^T PA & 0 & B^T PB + R - \alpha_2^{-2}R \end{bmatrix},$$

$$H_4 = \begin{bmatrix} A^T PA - \alpha_4^{-2}P & 0 & A^T PB \\ 0 & Q - \alpha_4^{-2}Q & 0 \\ B^T PA & 0 & B^T PB + R - \alpha_4^{-2}R \end{bmatrix}. \tag{28}$$

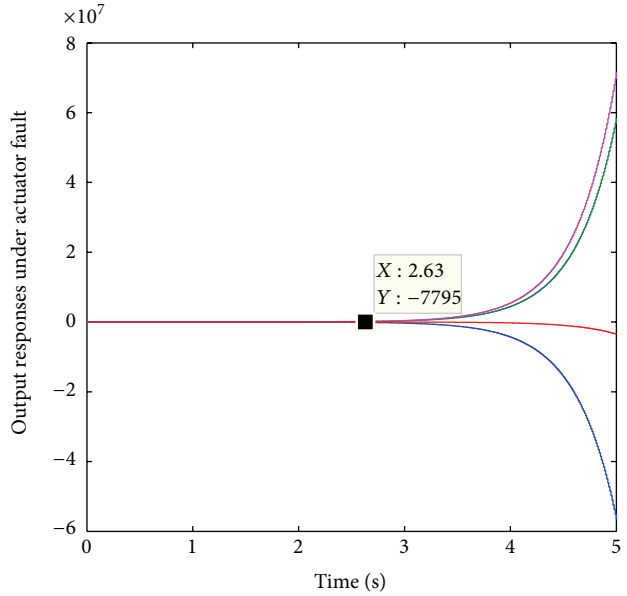


FIGURE 3: Output of the system with data packet dropout and actuator fault.

Similar to the above process, (14) and (16) can be obtained. Then, the following inequalities hold:

$$V(x_{k+1}) - \alpha_2^{-2} V(x_k) < 0,$$
$$V(x_{k+1}) - \alpha_4^{-2} V(x_k) < 0. \tag{29}$$

At last, form Lemma 1 and Theorem 3, the conclusion that the NCS (5) is exponentially stable with the decay rate δ is made. □

Remark 5. From Theorem 4, it can be found that the NCS (5) is exponentially stable. According to the characteristic of trivial solution of system (5), we can get that the exponential stability of NCS (5) is equivalent to asymptotic stability, which corresponds with the requirement of integrity design for FTC.

4. Numerical Examples

In this section, two numerical examples are given to illustrate the performance of the proposed approach.

Example 1. Consider the following linear model of a helicopter (as in [19]):

$$\dot{x}(t) = \overline{A}x(t) + \overline{B}u(t),$$
$$y(t) = \overline{C}x(t), \tag{30}$$

where $x = [V_h, V_v, q, \theta]^T$ is system state, $u = [\delta_c, \delta_l]^T$ is control input, V_h is horizontal velocity, V is vertical velocity, q is pitch rate, θ is angle of pitch, δ_c is blade control, and the matrices of model (30) are

$$\overline{A} = \begin{bmatrix} -0.0366 & 0.0271 & 0.0188 & -0.1555 \\ 0.0482 & -1.01 & 0.0024 & -4.0208 \\ 0.1002 & 0.3681 & -0.707 & 1.420 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$
$$\overline{B} = \begin{bmatrix} 0.4422 & 0.1761 \\ 3.5446 & -7.5922 \\ -5.52 & 4.49 \\ 0 & 0 \end{bmatrix}, \quad \overline{C} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{31}$$

Assume that the sampling period is $T = 0.1$ s and the initial state $x_0 = [250 \quad 50 \quad 10 \quad 8]^T$, then the matrices of zero-order holding system (30) are given by

$$A = \begin{bmatrix} 0.9964 & 0.002596 & 0.001039 & -0.01596 \\ 0.004513 & 0.9037 & -0.01879 & -0.3834 \\ 0.009762 & 0.03388 & 0.9383 & 0.1303 \\ 0.0004922 & 0.001741 & 0.09677 & 1.007 \end{bmatrix},$$
$$B = \begin{bmatrix} 0.04424 & 0.01687 \\ 0.3407 & -0.7249 \\ -0.5278 & 0.4214 \\ -0.02678 & 0.02151 \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}. \tag{32}$$

Suppose the rate of packet dropout is 30% in the S-C link and the rate of packet dropout is 20% in the C-A link; then, we have $r_1 = 0.7$ and $r_2 = 0.8$.

Denote actuator failure matrix L as follows:

$$L = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}. \tag{33}$$

From Figure 3, it can be seen that the system is unstable when the data packet dropout and actuator faults happen. Then, using the proposed approach in Theorem 4 and taking $\alpha_1 = 3.1641, \alpha_2 = 0.3461,$ and $\alpha_3 = 0.6121, \alpha_4 = 0.6188$, the controllers can be obtained as follows:

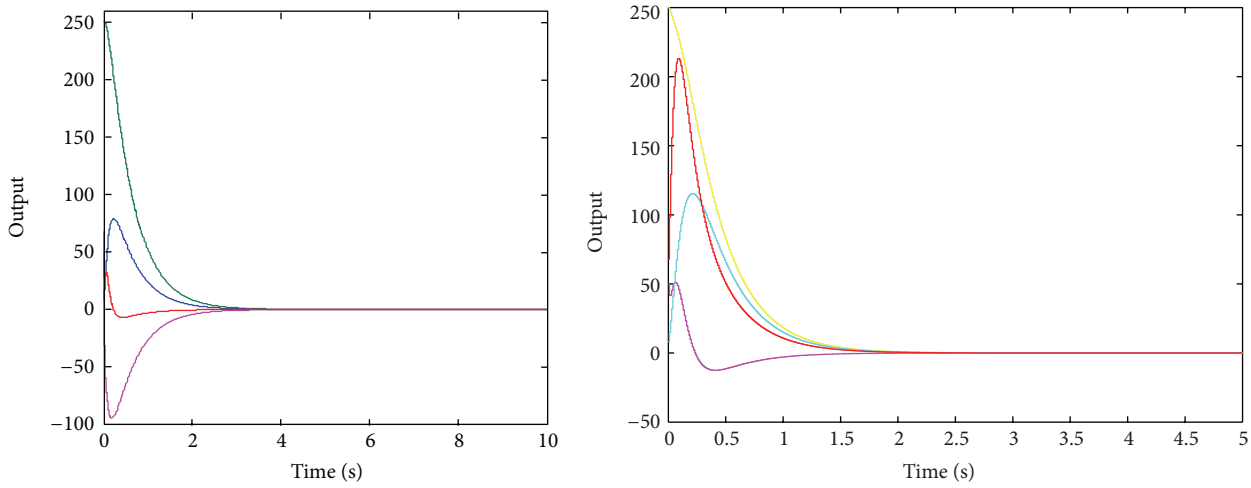
$$K_1 = \begin{bmatrix} 0.6129 & 0.3626 & -1.480 & -2.5049 \\ -0.3064 & -1.0651 & 0.2590 & 0.0393 \end{bmatrix},$$
$$K_3 = \begin{bmatrix} -1.1034 & -0.1149 & 0.8454 & 1.0289 \\ -0.1382 & 1.5734 & -0.1126 & -0.574 \end{bmatrix}. \tag{34}$$

Besides, from Figure 4, we can see that controllers can guarantee system stability when the data packet dropout and actuator faults happen. Therefore, the simulation result of Example 1 demonstrates the effectiveness of our presented method.

Example 2. Consider the discrete system as follows (as in [13]):

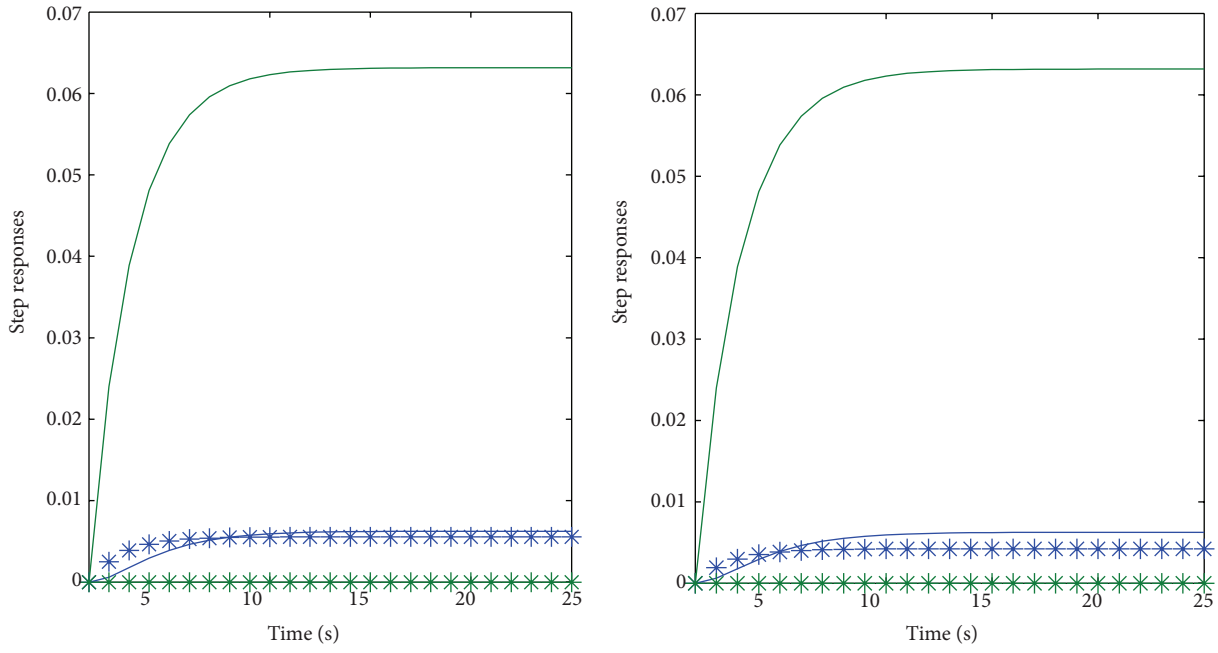
$$x(k+1) = \begin{bmatrix} 0.55 & 0.035 \\ 0 & 0.62 \end{bmatrix} x(k) + \begin{bmatrix} 0.006 \\ 0.24 \end{bmatrix} u(k),$$
$$y(k) = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix} x(k), \tag{35}$$

with the sampling period $T = 0.3$ s.



(a) Event 1 (b) Event 3

Figure 4: Output of the closed-loop NCS with data packet dropout and actuator fault.



(a) Event 1 (b) Event 3

Figure 5: Step responses under different discrete events.

When data packet dropouts occur in the communication channels, a controller presented in [13] can stabilize the closed-loop NCS. But, when the closed-loop NCS with actuator failure matrix $L = [1 \quad 0]$, the suitable controller and α_i cannot be found using method in [13]. Comparing with method in [13], for given $r_1 = 0.6, r_2 = 0.7, \alpha_1 = 1.574, \alpha_2 = 0.913, \alpha_3 = 0.705$, and $\alpha_4 = 0.72$, the suitable output feedback controllers can be found by the proposed approach in this paper:

$$K_1 = \begin{bmatrix} 0.0151 & 0.4880 \\ 0 & 0 \end{bmatrix}, \quad K_3 = \begin{bmatrix} 0.3171 & 0.0049 \\ 0 & 0 \end{bmatrix}. \tag{36}$$

Figure 5 shows the step responses of NCS (5) in events 1 and 3. The case without data packet dropout and actuator failure is displayed in Figure 5(a) as the solid lines, controllers can switch immediately between two different control loops, and the system is exponentially stable. When an actuator fault occurs, one of the trajectories becomes zero as shown in Figure 5(a) by the star parts. Similarly, Figure 5(b) shows that the proposed controllers can guarantee the stability even though both the data packet dropout and actuator failure happen. From a comparison between Figures 5(a) and 5(b), we can see that system performance in event 3 is not as good as that in event 1 because of the data packet dropout. Hence, the simulation results of Example 2 imply that the desired goal is well achieved.

5. Conclusion

The problem of FTC for NCSs with actuator failures and data packet dropout in both S-C and C-A links is discussed in this paper. These data packet dropouts are described by two independent switches, which can be modeled as a discrete event system with known rate. Introducing the matrix of actuator failures, the model of NCS with actuator failures is addressed as ADSs. Then, based on the theory of ADSs, the sufficiency of exponential stability for such NCS is derived and the output feedback controllers guaranteed system performance are presented. Finally, two numerical examples are exploited to show the effectiveness of the proposed method.

Acknowledgments

The authors would like to acknowledge the National Natural Science Foundation of P. R. China under Grant no. 61174044, the Independent Innovation Fund of Shandong University (2012JC005), and the Shandong Province Natural Science Foundation under Grant no. ZR2010FM016. The authors also wish to thank the reviewers for their valuable suggestions.

References

[1] W. Zhang, M. S. Branicky, and S. M. Phillips, "Stability of networked control systems," *IEEE Control Systems Magazine*, vol. 21, no. 1, pp. 84–97, 2001.

[2] G. C. Walsh, H. Ye, and L. G. Bushnell, "Stability analysis of networked control systems," *IEEE Transactions on Control Systems Technology*, vol. 10, no. 3, pp. 438–446, 2002.

[3] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 138–172, 2007.

[4] R. A. Gupta and M.-Y. Chow, "Networked control system: overview and research trends," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 7, pp. 2527–2535, 2010.

[5] M. B. G. Cloosterman, L. Hetel, N. van de Wouw, W. P. M. H. Heemels, J. Daafouz, and H. Nijmeijer, "Controller synthesis for networked control systems," *Automatica*, vol. 46, no. 10, pp. 1584–1594, 2010.

[6] L. Zhang, Y. Shi, T. Chen, and B. Huang, "A new method for stabilization of networked control systems with random delays," *IEEE Transactions on Automatic Control*, vol. 50, no. 8, pp. 1177–1181, 2005.

[7] Y. Shi and B. Yu, "Output feedback stabilization of networked control systems with random delays modeled by Markov chains," *IEEE Transactions on Automatic Control*, vol. 54, no. 7, pp. 1668–1674, 2009.

[8] S. Q. Wang, J. Feng, and H. G. Zhang, "Robust fault tolerant control for a class of networked control systems with state delay and stochastic actuator failures," *International Journal of Adaptive Control and Signal Processing*, 2012.

[9] C.-X. Yang, Z.-H. Guan, and J. Huang, "Stochastic fault tolerant control of networked control systems," *Journal of the Franklin Institute*, vol. 346, no. 10, pp. 1006–1020, 2009.

[10] E. Tian, D. Yue, and C. Peng, "Reliable control for networked control systems with probabilistic sensors and actuators faults," *IET Control Theory and Applications*, vol. 4, no. 8, pp. 1478–1488, 2010.

[11] C. Peng, T. C. Yang, and E. G. Tian, "Robust fault-tolerant control of networked control systems with stochastic actuator failure," *IET Control Theory and Applications*, vol. 4, no. 12, pp. 3003–3011, 2010.

[12] X. M. Huang, X. Li, C. J. Long, and Y. Gao, "Guaranteed cost fault-tolerant control of networked control systems with short output delay and short control delay based on state observer," *Journal of Networks*, vol. 8, no. 4, pp. 836–842, 2013.

[13] C. Wang, Z. Yuan, and Q. Li, "Stability of networked control systems with data packet dropout," in *Proceedings of the 4th IEEE Conference on Industrial Electronics and Applications (ICIEA '09)*, pp. 2734–2737, May 2009.

[14] X. M. Qi, C. Zhang, and J. Gu, "Robust fault-tolerant control for uncertain networked control systems with state-delay and random data packet dropout," *Journal of Control Science and Engineering*, vol. 2012, Article ID 734758, 7 pages, 2012.

[15] Z. Huo and H. Fang, "Research on robust fault-tolerant control for networked control system with packet dropout," *Journal of Systems Engineering and Electronics*, vol. 18, no. 1, pp. 76–82, 2007.

[16] M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control*, Springer, Berlin, Germany, 2003.

[17] J. Chen and R. J. Patton, *Robust Model-Based Fault Diagnosis for Dynamic Systems*, Kluwer Academic Publishers, Boston, Mass, USA, 1999.

[18] Y. Zhang and J. Jiang, "Bibliographical review on reconfigurable fault-tolerant control systems," *Annual Reviews in Control*, vol. 32, no. 2, pp. 229–252, 2008.

[19] B. Jiang, Z. H. Mao, H. Yang, and Y. M. Zhang, *Fault Diagnosis and Fault Accommodation for Control Systems*, National Defense Industry Press, 2009.

[20] S. X. Ding, P. Zhang, and Ch. Chihaia, "Advanced design scheme for fault tolerant distributed networked control systems," in *Proceedings of the 17th World Congress International Federation of Automatic Control*, pp. 13569–13574, 2008.

[21] Z. Mao, B. Jiang, and P. Shi, "Observer based fault-tolerant control for a class of nonlinear networked control systems," *Journal of the Franklin Institute*, vol. 347, no. 6, pp. 940–956, 2010.

[22] X. M. Qi, C. J. Zhang, and J. Gu, "Output feedback fault tolerent control for networked control systems with random access," *Control and Inteligent Systems*, vol. 40, no. 4, pp. 226–233, 2012.

[23] M.-Y. Zhao, H.-P. Liu, Z.-J. Li, and D.-H. Sun, "Fault tolerant control for networked control systems with packet loss and time

delay," *International Journal of Automation and Computing*, vol. 8, no. 2, pp. 244–253, 2011.

[24] A. Hassibi, S. P. Boyd, and J. P. How, "Control of asynchronous dynamical systems with rate constraints on events," in *Proceedings of the 38th IEEE Conference on Decision and Control (CDC '99)*, pp. 1345–1351, December 1999.

[25] A. Rabello and A. Bhaya, "Stability of asynchronous dynamical systems with rate constraints and applications," in *Proceedings of the American Control Conference*, pp. 1284–1289, May 2002.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 269398, 5 pages
http://dx.doi.org/10.1155/2013/269398

Research Article

Improved Weighted Shapley Value Model for the Fourth Party Logistics Supply Chain Coalition

Na Xu

School of Business, Shandong Jianzhu University, Jinan, Shandong 250101, China

Correspondence should be addressed to Na Xu; xuna_1011@163.com

Received 25 June 2013; Accepted 31 August 2013

Academic Editor: Jason Gu

How to make the individual get the reasonable and practical profit among the fourth party logistics supply chain coalition system is still a question for further study. Considering the characteristics of the fourth party logistics supply chain coalition, this paper combines Shapley Value with Distribution according to Contribution, two methods in the application, and then adjusts the profit allocated to each member reasonably based on the actual coalition situation named improved weighted Shapley Value model. In this paper, we first analyze the fourth party logistics supply chain coalition profit allocation models, the classical Shapley value method. Then, we analyze the weight of individual enterprise in the coalition by the analytic hierarchy process. To each enterprise, the weight is determined by the investment risks, information divulging risks, and failure risks. Finally, the numerical study shows that the profit allocation method improved weighted Shapley value model is relatively rational and practical. Thus, the proposed combined model is a useful profit allocation mechanism for the fourth party logistics supply chain coalition that the contribution and risks are fully considered.

1. Introduction

"The fourth party logistics (4PL) [1] is Supply Chain Integrated Provider, which integrate and manage the different resources, abilities and technologies belonging to the company's complementary service provider, and provide the overall solution to the supply chain with the customers." [2].

In this paper the fourth party logistics [3] particularly refers to the logistics and supply chain operation mode.

The fourth party logistics delivers supply chain service outsourcing to the fourth party, makes programs about the solutions to the supply chain management, and is responsible for the feedback of the supervision and management of solutions to supply chain management. Supply chain begins with the procurement of raw materials, and then, it produces intermediate products and final products and finally delivers products to consumers by sales system. It is the functional chain structure model consisting of manufacturers, distributors, retailers, and consumers.

The fourth party logistics supply chain refers to that the fourth party logistics service providers integrate and coordinate different types of resources, capabilities, and technologies belonging to competitive and complementary enterprises as shown in Figure 1. Its purpose is to integrate and optimize all the resources, technologies, and abilities of the enterprises in the coalition of the supply chain system. All these enterprises include suppliers, manufacturers, distributors, transporters, and warehousing and storage which are called logistics service providers and other auxiliary organizations. The fourth party logistics service providers play the roles of the builder and coordinator.

Risks arise when the fourth party logistics supply chain operates, because of the unpredicted effects brought by the unconfirmed factors of the inner and outside environment. The deviation between actual profit and expected profit and the possibility of the loss of the individual enterprise in the coalition can be enlarged [1–4].

The fourth party logistics supply chain risk management refers to identifying the risks existing when the supply chain operates, analyzing the mechanism, evaluating the effect, controlling and regulating all the risks by taking appropriate measures, and then enhancing the reliability of the supply chain operation through coalition among enterprises.

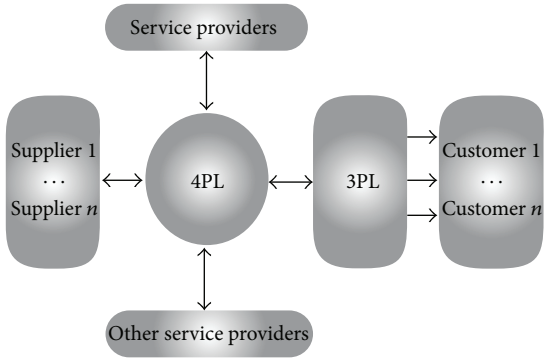


Figure 1: The fourth party logistics supply chain coalition.

The core enterprise of the supply chain knows the requirements of the market and coordinates with the other partners, and all the enterprises work together to make optima performance to improve the supply chain competiveness. The individual enterprise makes the individual profit maximum as well as the whole through the coalition. How to allocate profit rationally is determined by the contribution weight consisting of investment risks, information divulging risks, and information divulging risks.

Tanimoto has built models of the mechanism of cost allocation by the agents. The coalition and the rule of the cost allocation will be realized under the cost structure [5]. Moreover, Skorin-Kapov has suggested further analysis of the multicasting cost allocation problem using the modified multicasting game [6]. Also Chen et al. have studied the ratio of profit allocation based on the evolution game theory [7] and cooperative game theory [8].

A multiagent system model and game theory and Nash equilibrium algorithms based on market competition have been proposed for task allocation [9]. The supply chain profit has been allocated by using the risk-considered Shapley value method [10] and cooperation game theory [11–13]. Wang and Zhou have built profit allocation model of three-stage supply chain with multidirectional and principal subordinate [14].

The rest of the paper has been organized as follows: Section 2 states the problem of statement. Sections 3 and 4 propose the model and algorithm for profit allocation in the fourth party logistics supply chain coalition. A numerical study has been included in Section 5, and the last section concludes the paper.

2. Description of Problem

The relationship among member enterprises in the fourth party logistics supply chain coalition is a strategic collaboration partnership and in essence a competitive relationship based on cooperation. Based on the cooperation when the upstream and downstream enterprises compete with each other for the resources (including power resources) and thus enjoy the greater profits, the characteristics of the competitiveness exist.

When the member enterprises with the same level compete with each other for the orders, the characteristics of the competitiveness also exist. The relationship among member enterprises in the fourth party logistics supply chain coalition is collaborated. The member enterprises are not allowed to manage or to be managed. The member enterprises have their own decision power. The fourth party logistics service providers make suggestions for the work and profit allocation among the member enterprises according to the overall situation, then supervise the performing of the contract, and take punishment measures for breach of contracts. So the mode of the fourth party logistics supply chain coalition is mixed with the integration and separation [15].

The profit allocation is the major problem in the fourth party logistics supply chain coalition. Thus, how to allocate profit allocation rationally and practically among the member enterprises is the problem discussed in the paper.

The allocation of profits reasonably among the fourth party logistics supply chain coalition is the main objective of this paper.

3. Coalition Game

The main problem of the fourth party logistics supply chain coalition game is how to allocate the total output and utility reasonably. Thus, all the individuals can get the profit belonging to them.

Given the finite set N, coalition game is defined as $G = (N, V)$. Characteristic function v is the reflection from $2^N = \{S \mid S \subseteq N\}$ to real number set R^N, $v : 2^N \to R^N$, $v(\phi) = 0$:

$$v(S) = \sum_{i \in S} c(\{i\}) - c(S), \quad \forall S (S \subseteq N), \qquad v(\phi) = 0. \tag{1}$$

4. Profit Allocation Model

4.1. Shapley Value. Shapley Value is the classical method in the profit allocation and reflects the importance of the individuals among the fourth party logistics supply chain coalition enterprises according to the ratio of the value-added profit allocation. Shapley Value can be explained as the average marginal contribution in which each individual would make it to the grand coalition if it were to form an individual a time. Mathematically, Shapley Value is expressed as

$$y_i = \phi_i(v) = \sum_{i \in S} \frac{(n-s)!\,(s-1)!}{n!} \left(v(S) - v(S \setminus \{i\})\right), \quad \forall i \in N, \tag{2}$$

where s denotes the number of carriers in coalition S.

Shapley Value profit allocation method assumes the contribution among the enterprises in the alliance, not considering the risk factors, and so forth, in the allocation. Now the research about the improved Shapley Value based on the individual difference has been developed. Considering

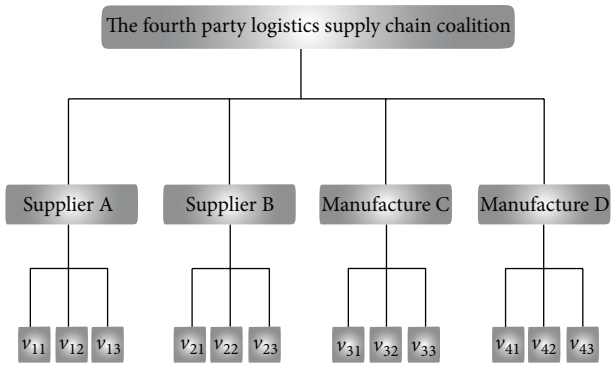


Figure 2: The hierarchical structure model.

the investment risks, information divulging risks, and failure risks, the different contribution among the coalition enterprises is different. So, we have been revising Shapley Value method [16–21].

The individual profit allocated by Shapley Value among the fourth party logistics supply chain coalition cannot be allocated according to the actual contribution and risks factors. So, we adjust the Shapley Value to make it apply to the actual situation and applications.

4.2. Distribution according to Contribution. Before we assume the contribution (including the risks) among the fourth party logistics supply chain coalition enterprises we should assume p_i. P_i denotes the weight of the importance of the coalition enterprises. To each coalition enterprise, the contribution weight is determined by the investment risks v_{i1}, information divulging risks v_{i2}, and failure risks v_{i3}.

P_i is determined by analytic hierarchy process (AHP). AHP is an effective and mature method. It can be used to deal with the complex relationships by quantitative analysis method. First, we decompose the complex relationship to different indexes, then make hierarchical structure diagrams according to the relationship, and finally build the judgment matrix according to the relative importance of the different layer indexes.

Here, we make the hierarchical structure model of the fourth party logistics supply chain coalition including object layer, rule layer, and schemes layer as shown in Figure 2.

We build the judgment matrix as shown in Table 1 according to the importance between the upper and lower layer on the basis of the hierarchical structure model. We adopt the 1–9 scale method by Professor Saaty. Among them 1 denotes the one that is as important as the other, 3 denotes the one that is slightly important than the other, 5 denotes the one that is much more important than the other, 7 denotes the one that is extremely important than the other, 9 denotes the one that is entirely important than the other [22–24], and x_i denotes the individual enterprise among the coalition.

According to the judgment matrix, we calculate the weight of the fourth logistics supply chain coalition enterprises by the square root method.

We first calculate the product of each row elements of the judgment matrix named $M_i = \prod_{j=1}^{n} x_{ij}$, then calculate the nth root of M_i named $p_i = \sqrt[n]{M_i}$, and finally normalize to calculate the weight of the fourth logistics supply chain coalition enterprises named

$$p_i = \frac{w_i}{\sum_{j=1}^{n} w_j}. \tag{3}$$

Table 1: Judgement matrix.

The fourth party logistics supply chain coalition enterprises	x_1	x_2	x_3	x_4
x_1	1	7	3	5
x_2	1/7	1	1/6	1/2
x_3	1/3	6	1	4
x_4	1/5	2	1/4	1

Table 2: Weight of coalition.

The fourth party logistics supply chain coalition enterprises	Weight
p_1	0.56
p_2	0.057
p_3	0.29
p_4	0.097

Because we cannot assure the complete consistency of the judgment matrix when the elements are compared with each other, we need to verify the complete consistency. Calculate

$$\lambda_{\max} = \frac{1}{n}\left(\sum_{i=1}^{n} \frac{\sum_{j=1}^{n} x_{ij} p_j}{p_i}\right) = 4.13, \tag{4}$$

where $\lambda_{\max}$ is the biggest characteristic value of the judgment matrix. The consistency index value is CI $= (\lambda_{\max} - n)/(n - 1) = 0.042$. Random consistency ratio is CR $=$ CI/RI < 0.1. The average random consistency index RI is 0.89 when the order number is 4. Because CR < 0.1, it is in accordance with random consistency.

5. Analysis

Now we have the fourth party logistics supply chain coalition which has two suppliers and two manufacturers named l_i. We use the Shapley Value and the distribution according to the contribution for the fourth party logistics supply chain coalition.

Data of the operation cost and profit of individual enterprises independent operation are shown as follows. (The test data is from a company.)

There are four enterprises in the fourth party logistics supply chain coalition. We calculate the cost and profit after the coalition. We get cost savings of coalition using (1). We analyze the weight of the enterprises in the fourth party logistics supply chain coalition as shown in Table 2.

We use vector (enterprises, profit, cost, and contribution to coalition) to show the instance.

The data can be expressed by vector (enterprises, profit, cost, and contribution to coalition).

The data is shown as follows:

$(x_1, 0, 5603.7, 26557.6)$ $(x_2, 0, 4156.9, 19197.6)$
$(x_3, 0, 4598.4, 24976.6)$ $(x_4, 0, 5406.9, 19909.6)$
$(x_1\ x_2, 1216.7, 8543.9, 0)$ $(x_1\ x_3, 1768.3, 8433.8, 0)$
$(x_1\ x_4, 1174.0, 9836.6, 0)$ $(x_2\ x_3, 975.3, 7780, 0)$
$(x_2\ x_4, 629.7, 8934.1, 0)$ $(x_3\ x_4, 1516.9, 8488.4, 0)$
$(x_1\ x_2\ x_3, 2770.6, 11588.4, 0)$
$(x_1\ x_2\ x_4, 2772.9, 12394.6, 0)$
$(x_1\ x_3\ x_4, 3123.6, 12485.4, 0)$
$(x_2\ x_3\ x_4, 2881.1, 11281.1, 0)$
$(x_1\ x_2\ x_3\ x_4, 4527.5, 15238.4, 0)$.

We calculate the weight using (3):

$$(p_1, p_2, p_3, p_4) = (0.56, 0.057, 0.29, 0.097). \quad (5)$$

The profit allocated to each individual in the fourth party logistics supply chain coalition that can be calculated using the Shapley Value Equation (2) is shown as follows:

$$(x_1\ x_2\ x_3\ x_4) = (1122.45, 916.57, 1272.43, 1118.22). \quad (6)$$

The profit allocated to each individual in the coalition can be calculated using Distribution according to Contribution and is shown as follows:

$$(x_1\ x_2\ x_3\ x_4) = (2480.6, 252.5, 1284.6, 429.7). \quad (7)$$

In conclusion, we can see that individual C gets the maximum profit according to Shapley Value while individual A gets the maximum profit according to Distribution according to Contribution and adjusts the two individuals AC which get the maximum profits and adjust the individuals BD reasonably in the same way. The value should be adjusted in accordance with the needs of the coalition in order to make the coalition stronger and better. We should combine the two methods named the improved weighted Shapley Value model for the practical use.

6. Conclusion

First of all, we introduce the fourth party logistics and the fourth party logistics supply chain coalition. How to make profit allocation practically and reasonably among the coalition is still a question for further study. We can also call this problem the fourth party logistics supply chain coalition game.

Besides, we introduce the classical method, Shapley Value. We consider the investment risks, information divulging risks and failure risks besides the only contribution. On the basis of the risks, we analyze the weight of the importance of individuals in the coalition by the analytic hierarchy process. To each enterprise, the weight is determined by the investment risks, information divulging risks, and failure risks. We calculate the weight of the fourth party logistics supply chain and assure the complete consistency of the judgment matrix when the elements are compared with each other.

Finally, we analyze the results using Shapley Value and the Distribution according to Contribution. We find that we should combine the two methods for the practical use. The value should be adjusted in accordance with the needs of the coalition in order to make the coalition stronger and better.

We also need further improvement of the two combined methods named the improved weighted Shaplye Value model. We also need further study of how much profit to adjust which is allocated by Shapley Value and contribution weight (including the risks) based on the actual situation.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is sponsored in part by a grant from the National Science Foundation (NSF) (project Grant no. 61202363).

References

[1] Y. Peng, *The research of the fourth logistics supply chain management [Ph.D. thesis]*, Tianjin University, Tianjin, China, 2006.

[2] C. Liu, *Fourth party logistics optimization automotive supply chain logistics [M.S. thesis]*, Jilin University, Jilin, China, 2009.

[3] Q. Zhu and R. Y. K. Fung, *Design and Analysis of Optimal Incentive Contracts between Fourth-Party and Third-Party Logistics Providers*, System Engineering and Engineering Management, City University of Hong Kong, Hong Kong, 2012.

[4] D. Wang, *The research of supplier selection and profit allocation under supply chain risk [Ph.D. thesis]*, Dalian Maritime University, Liaoning, China, 2007.

[5] K. Tanimoto, "Bargaining over cost allocation rule under uncertainty of project cost," in *Proceedings of the 2001 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1270–1275, 2001.

[6] D. Skorin-Kapov, "On primal-dual cost allocation schema in multicast communication: combinatorial game theory model," in *Proceedings of the IEEE 17Th International Conference on Industrial Engineering and Engineering Management (IE&EM '10)*, pp. 433–438, 2010.

[7] Z. Chen, S. Yang, and Y. Cao, "Profit allocation in mobile commerce value chain based on the evolution game theory," in *Proceedings of the 2nd IEEE International Conference on Information and Financial Engineering (ICIFE '10)*, pp. 296–300, 2010.

[8] E. Semsar-Kazerooni and K. Khorasani, "A game theory approach to multi-agent team cooperation," in *Proceedings of the American Control Conference*, pp. 4512–4518, 2009.

[9] G. Wang, H. Yu, J. Xu, and S. Huang, "A multi-agent model based on market competition for task allocation: a game theory approach," in *Proceedings of the IEEE International Conference on Networking, Sensing and Control*, pp. 282–286, 2004.

[10] R. Sui and F. Li, "Risk-considered Shapley profit allocation of innovative supply chain," in *Proceedings of the IEEE International Conference on Emergency Management and Management Sciences (ICEMMS '10)*, pp. 238–241, 2010.

[11] L. Chen, M. Shen, and C. Chen, "A research in supply chain profit allocation based on cooperation game theory," in *Proceedings of the International Conference on System Science, Engineering Design and Manufacturing Informatization (ICSEM '10)*, pp. 209–212, 2010.

[12] N. Matsubayashi, M. Umezawa, Y. Masuda, and H. Nishino, "A cost allocation problem arising in Hub-Spoke network systems," *European Journal of Operational Research*, vol. 160, no. 3, pp. 821–838, 2005.

[13] D. J. Wan, Y. Liu, and B. Yao, "Research on cost allocation in the cooperative game of overflowing infrastructure investment," in *Proceedings of the International Conference on Management Science and Engineering*, pp. 393–398, 2007.

[14] L. Wang and Y. Zhou, "Research on cooperation profit allocation in three-stage supply chain based on distribution channel," in *Proceedings of the 4th International Conference on Wireless Communications, Networking and Mobile Computing*, pp. 1–6, 2008.

[15] H. Goudarzi and M. Pedram, "Maximizing profit in cloud computing system via resource allocation," in *Proceedings of the International Conference on Distributed Computing Systems Workshops*, pp. 3–4, University of Southern California, 2011.

[16] H. Cheng and G. Yang, "A study on profit allocation model based on integrated supply," in *Proceedings of the International Conference on Services Systems and Services Management (ICSSSM '05)*, pp. 613–616, June 2005.

[17] S. Wang and B. Zhao, "Study on profit allocation among green supply chain based on the optimized shapely value," in *Proceedings of the 3rd International Conference on Information Management, Innovation Management and Industrial Engineering (ICIII '10)*, pp. 252–255, November 2010.

[18] N. Matsubayashi, M. Umezawa, Y. Masuda, and H. Nishino, "A cost allocation problem arising in hub-spoke network systems," *European Journal of Operational Research*, vol. 160, no. 3, pp. 821–838, 2005.

[19] G. Owen, "On the core of linear production games," *Mathematical Programming*, vol. 9, no. 1, pp. 358–370, 1975.

[20] L. S. Shapley, "A value for n-person games," *Annals of Mathematical Studies*, vol. 28, pp. 307–317, 1953.

[21] H. P. Young, "Cost allocation: methods, principles, applications," *European Journal of Operational Research*, vol. 27, no. 2, pp. 254–255, 1986.

[22] L. Saaty, *The Analytic Hierarchy Process*, McGraw-Hill Company, New York, NY, USA, 1980.

[23] L. Yi, *Study on local fiscal risk and its measure to keep a way in China [Ph.D. thesis]*, Chongqing University, Chongqing, China, 2005.

[24] H. Yuan, "The construction and empirical analysis of nonparametric early-warning system of fiscal risk based on risk factor method and AHP," *Journal of Hebei University of Economics and Business*, vol. 6, pp. 5–8, 2011.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 416715, 13 pages
http://dx.doi.org/10.1155/2013/416715

Research Article

Mobile Robot Path Planning Using Polyclonal-Based Artificial Immune Network

Lixia Deng,[1] Xin Ma,[1] Jason Gu,[1,2] and Yibin Li[1]

[1] *School of Control Science and Engineering, Shandong University, Jinan, Shandong 250061, China*
[2] *Department of Electrical and Computer Engineering, Dalhousie University, Halifax, NS, Canada B3J 2X4*

Correspondence should be addressed to Xin Ma; maxin@sdu.edu.cn and Jason Gu; jason.gu@dal.ca

Received 14 June 2013; Accepted 14 August 2013

Academic Editor: Fei Liu

Polyclonal based artificial immune network (PC-AIN) is utilized for mobile robot path planning. Artificial immune network (AIN) has been widely used in optimizing the navigation path with the strong searching ability and learning ability. However, artificial immune network exists as a problem of immature convergence which some or all individuals tend to the same extreme value in the solution space. Thus, polyclonal-based artificial immune network algorithm is proposed to solve the problem of immature convergence in complex unknown static environment. Immunity polyclonal algorithm (IPCA) increases the diversity of antibodies which tend to the same extreme value and finally selects the antibody with highest concentration. Meanwhile, immunity polyclonal algorithm effectively solves the problem of local minima caused by artificial potential field during the structure of parameter in artificial immune network. Extensive experiments show that the proposed method not only solves immature convergence problem of artificial immune network but also overcomes local minima problem of artificial potential field. So, mobile robot can avoid obstacles, escape traps, and reach the goal with optimum path and faster convergence speed.

1. Introduction

Robot path planning is one of the most fundamental functions for mobile robot. There are two types of planning for autonomous mobile robot based on how much information is known about static environment: global path planning when the environment is clearly known and sensory-based local path planning when partial or no information about the environment is known in advance [1]. In the global path planning, the environment surrounding the robot and the position of obstacles are well known in advance, and the robot is required to navigate to its destination with avoiding obstacles. The complete robot path in such applications can be calculated from the prior knowledge of the coordinates of the starting point, the destination point, and obstacles. On the other hand, the local motion planning dynamically guides the robot according to the locally sensed obstacles, which requires less prior knowledge about the environment. Therefore, the local path planning methods are more suitable and practical for mobile robot, since the environment is too complicated to be known precisely and may also be time-varying [2]. While, the local path planning faces significant challenges.

This paper focuses on mobile robot path planning in unknown complex static environment. Artificial immune network (AIN) can optimize mobile robot path planning, but it exists as a problem of immature convergence. So polyclonal-based artificial immune network (PC-AIN) is proposed to solve the problem. Immunity polyclonal algorithm (IPCA) increases the diversity of antibodies which tend to the same extreme value through clonal operator, clonal crossover operator, clonal mutation operator and clonal selection operator, and finally selects the antibody with highest concentration. Meanwhile, immunity polyclonal algorithm solves the problem of local minima caused by artificial potential field during the structure of parameter in

artificial immune network. Experimental results show that the proposed method effectively optimizes path planning and improves the performance of path planning.

The remainder of this paper is organized as follows. The next section discusses related works for autonomous mobile robot path planning in unknown complex environment. Section 3 presents the proposed mobile robot path planning algorithm in detail. Simulation results and real-time experiments are presented in Section 4. Finally, Section 5 concludes the paper.

2. The Related Works

Local path planning in unknown complex environment is a hot research topic in recent years. Artificial potential field-based path planning algorithm [3, 4] utilizes the attractive force from goals and the repulsive force from obstacles for mobile robot path planning. This method is known for its mathematical simplicity and little computation, but it suffers from a problem of local minima. Many artificial intelligent techniques have been proposed by scholars for mobile robot path planning, such as reinforcement learning, fuzzy logic, and genetic algorithm. Reinforcement learning algorithm [1, 5, 6] has simple and complete theory, but it is mostly used in static environment because its infinite state in complex environment. Fuzzy logic control (FLC) [7–9] has the capacity to handle uncertain and imprecise information obtained from sensors using linguistic rules. The advantages of fuzzy system are that it does not require precise analytical models of environment and it has better real-time performance, but its computational complexity increases with the geometric progression growth of rules' number. Genetic algorithm (GA) [10, 11] is a multipoint searching algorithm, and it is more likely to search the global optimal solution in static and dynamic environment. However, GA suffers from immature convergence in complex environment.

Artificial immune system (AIS) is an optimization algorithm based on the biological immune system. It has the following features: self-organization, intelligence, recognition, adaptation, and self-learning. There are lots of researches investigating the interactions between various components of the immune system or the overall behaviors of systems based on an immunological point of view. Artificial immune system mainly includes the following categories.

Innate immune-based path planner: Deepak et al. [12] proposed a motion planner motivated from the biological innate immune system. To actuate a suitable robotic action, one new parameter named as learning rate has been introduced. The further movement of robot is decided by selecting of a suitable robot predefined task, so the robot will move in sequence until it reaches to its destination. Robot actions are defined as antibodies and environmental scenarios are defined as antigens. The algorithm is simple and less numerical complex because of very few controlling parameters in this structure.

Immune genetic algorithm: Chen et al. [13] proposed a new method of optimal path planning for mobile robots based on immune genetic algorithm with elitism. The grid theory is utilized to establish the free space model of the mobile robot in a given environment, and a sequence number is used to identify a grid. The initial chromosomes are generated with a string of sequence numbers. Meanwhile, an insertion operator and a deletion operator are defined to ensure that each obtained path is continuous and collision-free.

Jerne's idiotypic immune network: many researchers utilize the equation generated by Farmer et al. based on the hypothesis proposed by Jerne (1973) for path planning. (1) *Immune network model:* Raza and Fernandez [14] used the dynamic equation of the idiotypic system to perform searching and rescuing operation in unstructured environment. The environment is translated into antigen according to the sensory arrangement of every robot. Antibody stimulation and suppression are implemented to navigate robots along with the idiotypic connections to establish interrobot communications. Yuan et al. [15] proposed an improved artificial immune network strategy based on APF method to avoid absolute random transition of antibody and improve the searching efficiency of immune network. The environment surrounding the robot is defined as antigen and robot action is defined as antibody, respectively. The planning results of APF method is regarded as prior knowledge, and the instruction definition of new antibody is initialized through vaccine extraction and inoculation. (2) *Reactive immune network:* Luh and Liu [16] proposed reactive immune network algorithm for mobile robot path planning with data representation for antigens and steering directions for antibodies in the U-shape environment. In [16], the resultant force of artificial potential field is defined as the affinity between antigen and antibody. Adaptive virtual target method is used to solve the local minima problem of artificial potential field method. (3) *Mixture of artificial immune algorithm:* Challoo et al. [17] proposed a mixture algorithm based on immune network and negative selection. The introduction of negative selection makes the system faster to react for certain conditions which are defined as self-conditions. After computing the concentration of antibodies, there is a reward or a penalty to train an antibody incrementally. The proposed controller provides a principled way for organizing an intelligent robotic system. Moreover, implementation of this controller is done in real time in three different scenarios. Ozcelik and Sukumaran [18] used the dynamic equation of the idiotypic system as output and the concentration of helper T cells as feedback system to control the output. The squashing function is performed by awarding or penalizing the antibody based on the antibody that has the highest priority. The highest priority is given to the antibody which perfectly matches to the antigen. The concentration of helper T cells is used to compute the number of penalties computing.

But the existing artificial immune network algorithm exists as a problem of immature convergence. The ability of optimizing navigation path is decreased with the decreasing of antibody diversity.

Clonal selection algorithm [19–26] is a new algorithm of artificial immune algorithm, which is designed based on the

adaptive immune clonal selection principle and is widely used in pattern recognition, control, optimization, multimodal, and combination problem.

Clonal selection: Cortes and Coello [22] introduced a new multiobjective optimization approach based on the clonal selection principle. They indicate that the use of an artificial immune system for multiobjective optimization is a viable alternative. Huizar et al. [24] used the clonal selection algorithm to generate optimal or nearly optimal solutions for solving combinatorial optimization problems. Meng and Qiuhong [23] proposed a new artificial immune algorithm based on the clonal selection theory and the structure of anti-idiotype (IAAI). IAAI is improved to achieve the evolution of the whole antibody population, and the new algorithm has strong searching capability which makes it reach better performance by performing global search and local search in many directions in the solution space.

Monoclonal algorithm: Liu et al. [25] proposed Immune Monoclonal Strategy Algorithm (IMSA) which realizes the global optimal computation as well as the local search. According to antibody-antigen affinity, the algorithm adaptively adjusts the clonal scale of antibody population. Moreover, it shows that IMSA has the strong abilities in having high convergence speed, enhancing the diversity of the population, and avoiding the immature convergence to some degree. However, the algorithm has the problem of weakening local searching ability because of that the local searching mechanism of the algorithm itself is not perfect and the optimization function is complicated.

Polyclonal algorithm: ploy clonal is the foundation of the specific immune response. Different from the monoclonal strategy which has only one or a few antigenic determinant and epitope, polyclonal performance at the cellular level is the structure of extreme diversity of TCR (T cell antigen receptors) and BCR (B cell antigen receptor). Therefore, it can directly lead to the diversity of antibody network, memory, and specificity. Shen and Yuan [26] proposed a novel polyclone particle swarm optimization algorithm (PCPSOA) for mobile robot path planning. PCPSOA is characterized by strong searching ability and quick convergence speed. The simulation results show that the path planning based on PCPSOA is feasible and effective.

Immunity polyclonal algorithm is used to solve the immature convergence problem of artificial immune network and the local minima problem of artificial potential field method. The proposed polyclonal based artificial immune algorithm increases diversity of antibodies which tend to the same extreme value in the solution space of artificial immune network using clonal operator, clonal crossover operator, clonal mutation operator, and clonal selection operator.

3. The Proposed Path Planning Algorithm

3.1. Graphical Representation of the Proposed Algorithm. In this paper, polyclonal-based artificial immune network (PC-AIN) is used for mobile robot path planning in unknown static environment. Artificial immune network (AIN) is used to compute concentrations of antibodies. Immunity polyclonal algorithm (IPCA) is used to solve the problems of immature convergence and local minima. The graphical representation of the proposed path planning system is depicted in Figure 1.

3.2. Antigen and Antibody Representation. Artificial immune network algorithm transforms the actual state of robot and defines antibodies and antigens from the perspective of information processing. Antigen's epitope is a data set detected by sensors and provides the information about the relationship among the current positions of robot, obstacles, and goal. An antigen may have several different epitopes, which means that an antigen can be recognized by a number of different antibodies. However, an antibody can bind only one antigen's epitope. In this paper, a paratope with a built-in robot's steering direction is regarded as antibody. These antibodies/steering directions are induced by recognition of the available antigens/detected information. It should be noted that only the antibody with highest concentration will be selected to act on robot according to the immune network hypothesis. Antigen represents the local environment surrounding the robot, and its epitope is a data set containing the azimuth of the goal θ_g, the distance between obstacles and the jth sensor d_j, and the azimuth of sensor θ_{S_j}:

$$A_{g_j} \equiv \left\{\theta_g, d_j, \theta_{S_j}\right\}, \quad j = 1, 2, \ldots, N_S, \tag{1}$$

where N_S is the number of sensors equally spaced around the base plate of the mobile robot, $d_{\min} \le d_j \le d_{\max}$. Parameters $d_{\min}$ and $d_{\max}$ represent the nearest and longest distance measured by the range sensors, respectively. $\theta_{S_j} = 2\pi(j-1)/N_S$, $j = 1, 2, \ldots, N_S$.

In addition, the relationship between antibody and steering direction is illustrated as follows:

$$\mathrm{Ab}_i \equiv \theta_i = \frac{2\pi}{N_{\mathrm{Ab}}}(i-1), \quad i = 1, 2, \ldots, N_{\mathrm{Ab}}, \tag{2}$$

where N_{Ab} is the number of antibodies and θ_i is the steering angle between the moving path and the head orientation of the mobile robot. Note that $0 \le \theta_i \le 2\pi$. There is no necessary relationship between N_{Ab} and N_S since they depend on the hardware of mobile robot. Nevertheless, simulation results show that better performance could be derived if N_S equal to or larger than N_{Ab}. In this paper, $N_S = N_{\mathrm{Ab}} = 8$.

3.3. Artificial Immune Network. Immune response forms a dynamic balance network through the interaction between antibody and antigen and the interaction between antibodies. When antigens invade the body, it achieves a new balance through the regulation of the immune system. Even when no antigen intrusion, it maintains the appropriate intensity through mutual stimulation and suppression between antibodies and it conforms to the idiotypic network hypothesis. In this paper, this reaction mechanism is adopted. The dynamic equation proposed by Farmer et al. is employed to

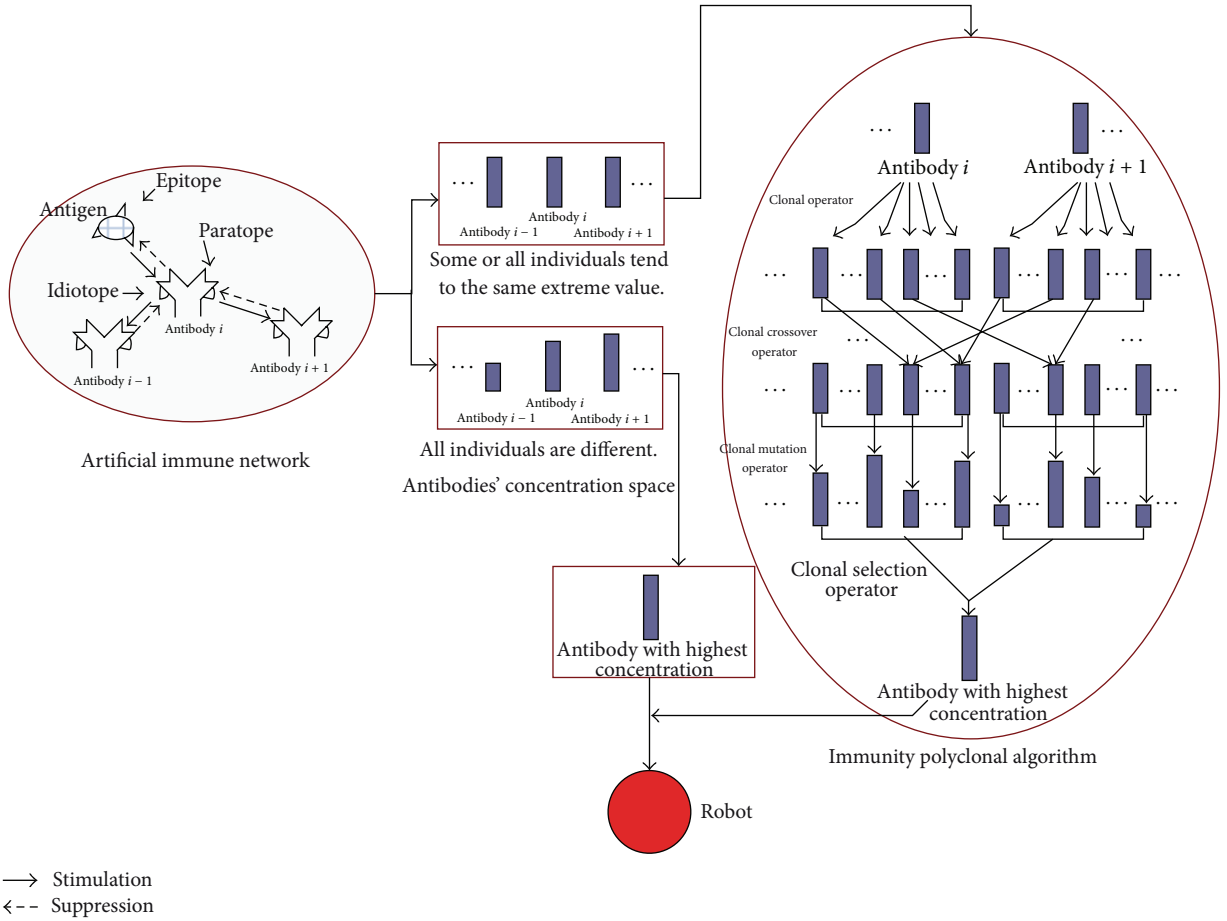


Figure 1: The graphic representation of the proposed path planning system. Antigen represents the local environment surrounding the robot. Antigen's epitope represents a data set detected by sensors including the azimuth of the goal, the distance between obstacles and sensor, and the azimuth of sensor. Antibody represents robot's steering direction. The interaction between antibodies is achieved by idiotope and paratope. Only the antibody with highest concentration will be selected to act on robot.

calculate the variation of antibody concentration as shown in the following equations:

$$\frac{dA_i(t)}{dt} = \left(\sum_{j=1}^{N_{\text{Ab}}} m_{ij}^{\text{st}} a_j(t) - \sum_{k=1}^{N_{\text{Ab}}} m_{ki}^{\text{su}} a_k(t) + m_i - k_i \right) \times a_i(t), \quad (3)$$

$$a_i(t) = \frac{1}{1 + \exp\left(0.5 - A_i(t)\right)}, \quad (4)$$

where $i, j, k = 0, 1, \ldots, N_{\text{Ab}}$ are the subscripts to distinguish the antibody types and N_{Ab} is the number of antibodies. a_i is the concentration of the ith antibody and dA_i/dt is the rate of change of concentration. m_{ij}^{st}, m_{ki}^{su} indicate the stimulative and suppressive affinity between the ith and the jth, kth antibodies, respectively. m_i denotes the affinity between antigen and the ith antibody, and k_i represents the natural death coefficient. Equation (3) is composed of four terms. The first term shows the stimulative interaction between antibodies, while the second term depicts the suppressive interaction between antibodies. The third term is the stimulus from the antigen, and the final term is the natural extinction term, which indicates the dissipation tendency in the absence of any interaction. Equation (4) is a squashing function to ensure the stability of the concentration [16].

The affinity between antigen and the ith antibody m_i is computed using artificial potential field method. In the proposed immune network, the resultant force of artificial potential field is defined as m_i, and the calculating process of m_i is illustrated as follows.

3.3.1. Attractive Force of the ith Antibody/Steering Direction. Consider

$$F_{\text{goal}_i} = \frac{1.0 + \cos\left(\theta_i - \theta_g\right)}{2.0}, \quad i = 1, 2, \ldots, N_{\text{Ab}}, \quad (5)$$

where $0 \le F_{\text{goal}_i} \le 1$. Obviously, the attractive force is at its maximal level ($F_{\text{goal}_i} = 1$) when the mobile robot goes straightforward to the goal ($\theta_i - \theta_g = 0$). On the contrary, it is minimized ($F_{\text{goal}_i} = 0$) when the robot's steering direction is the opposite of the goal ($\theta_i - \theta_g = \pi$).

3.3.2. Repulsive Force of the ith Antibody/Steering Direction. Consider

$$F_{\text{obs}_i} = \sum_{j=1}^{N_S} \alpha_{ij} \overline{d}_j, \tag{6}$$

where $\alpha_{ij} = \exp(-N_S \times (1 - \delta_{ij}))$ and $\delta_{ij} = (1 + \cos(\theta_i - \theta_{S_j}))/2$. Parameter α_{ij} indicates the weighting ratio for the jth sensor to steering angle θ_i, while $\overline{d}_j$ represents the normalized distance between the jth sensor and obstacles. Coefficient δ_{ij} expresses influence and importance of each sensor at different locations. $\overline{d}_j$ is fuzzified using the fuzzy set definitions. Three fuzzy if-then rules are defined to compute $\overline{d}_j$:

$$\begin{aligned} &\text{if } d_j \text{ is } s, \quad \text{then } y = L_1, \\ &\text{if } d_j \text{ is } m, \quad \text{then } y = L_2, \\ &\text{if } d_j \text{ is } d, \quad \text{then } y = L_3, \end{aligned} \tag{7}$$

where L_1, L_2, and L_3 are defined as 0.25, 0.5, and 1.0, respectively. The input variable of each rule is the detected distance d_j of the jth sensor. *s, m,* and *d* represent *safe*, *medium*, and *danger*, respectively. The membership function of *s, m,* and *d* are defined as (8), (9), and (10), respectively

$$\mu_s = \begin{cases} 0, & d_j \le d_m, \\ \dfrac{d_j - d_m}{d_{\max} - d_m}, & d_m < d_j \le d_{\max}, \\ 1, & d_j > d_{\max}, \end{cases} \tag{8}$$

$$\mu_m = \begin{cases} 0, & d_j \le d_{\min}, \\ \dfrac{d_j - d_{\min}}{d_m - d_{\min}}, & d_{\min} < d_j \le d_m, \\ \dfrac{d_{\max} - d_j}{d_{\max} - d_m}, & d_m < d_j \le d_{\max}, \\ 0, & d_j > d_{\max}, \end{cases} \tag{9}$$

$$\mu_d = \begin{cases} 1, & d_j \le d_{\min}, \\ \dfrac{d_m - d_j}{d_m - d_{\min}}, & d_{\min} < d_j \le d_m, \\ 0, & d_j > d_m. \end{cases} \tag{10}$$

The normalized distance between jth sensor and obstacles $\overline{d}_j$ is computed as (11)

$$\overline{d}_j = \frac{\mu_s \times L_1 + \mu_m \times L_2 + \mu_d \times L_3}{\mu_s + \mu_m + \mu_d}. \tag{11}$$

3.3.3. Resultant Force of the ith Antibody/Steering Direction. Consider

$$m_i = \omega_1 F_{\text{goal}_i} + \omega_2 F_{\text{obs}_i}, \quad i = 1, 2, \ldots, N_{\text{Ab}}, \tag{12}$$

where parameters ω_1, ω_2 indicate the weighting ratio between attractive and repulsive forces, $0 \le \omega_1, \omega_2 \le 1$, $\omega_1 + \omega_2 = 1$.

The stimulative and suppressive affinity between antibodies are combined and the stimulative-suppressive affinity between antibodies m_{il}^{ss} is defined as (13)

$$\begin{aligned} m_{il}^{\text{ss}} = m_{il}^{\text{st}} - m_{li}^{\text{su}} = \cos(\theta_i - \theta_l) = \cos(\Delta\theta_{il}), \\ i, l = 1, 2, \ldots, N_{\text{Ab}}. \end{aligned} \tag{13}$$

Obviously, stimulative-suppressive effect is positive ($m_{il}^{\text{ss}} > 0$) if $0 < \Delta\theta_{il} < \pi/2$, or $3\pi/2 < \Delta\theta_{il} < 2\pi$. On the contrary, stimulative-suppressive effect is negative ($m_{il}^{\text{ss}} < 0$) if $\pi/2 < \Delta\theta_{il} < 3\pi/2$. In addition, there is no any net effect between orthogonal antibodies (i.e., $\Delta\theta_{il} = \pi/2$, or $\Delta\theta_{il} = 3\pi/2$).

Equations (3) and (4) are discretized:

$$\begin{aligned} A_i(n+1) &= A_i(n) \\ &\quad + \left(\sum_{j=1}^{N_{\text{Ab}}} m_{ij}^{\text{st}} a_j(n) - \sum_{k=1}^{N_{\text{Ab}}} m_{ki}^{\text{su}} a_k(n) + m_i - k_i \right) \\ &\quad \times a_i(n), \\ a_i(n+1) &= \frac{1}{1 + \exp(0.5 - A_i(n+1))}, \end{aligned} \tag{14}$$

where n is the initial time at every step and $n+1$ is the current time at every step.

3.4. Immunity Polyclonal Algorithm. Artificial immune network exists as a problem of immature convergence which some or all individuals in the solution space tend to the same extreme value and robot will stop moving or move with wrong direction. From the view of biology, the reason of immature convergence problem is that the antibody concentration increases exponentially and some or all individuals tend to the same extreme value, so the diversity of antibody is decreased, and one antigen's epitope can be recognized by several antibodies at the same time. The phenomenon does not conform with the biological immune theory which one epitope can only be identified by one antibody. Specific to the process of mobile robot path planning, the normalized distance between sensor and obstacles reaches the maximum value 1 when the distance between robot and obstacles is less than the risk threshold, and concentrations of some or all antibodies tend to the same extreme value. Robot selects the antibody with highest concentration to act on at every step, but at the above situation, the moving probabilities toward certain directions are equal, and the robot has multiple optimal solutions at this point. So the robot will act wrongly.

Artificial potential field method is applied to compute parameter in artificial immune network, so artificial immune network exists as a problem of local minima. Attractive force and repulsive force have effect on robot, and there are some points which balance the attractive force and repulsive force, so the resultant force is zero at these points. It is the cause of local minima. More densely obstacles and more repulsive forces act on robot, so greater danger of local minima. When

the robot traps in local minima, robot will shock or stop at this point, and this phenomenon is known as "dead lock".

When the problems of immature convergence and local minima appear in the process of mobile robot path planning, robot periodically wanders around at the point and cannot reach the goal successfully. In this paper, immunity polyclonal algorithm is utilized to solve the above two problems.

The essence of immunity polyclonal algorithm is creating a new subpopulation based on the value of affinity around the candidate solution in the generation of evolution and expanding the searching scope. Meanwhile, immunity polyclonal algorithm implements the exchange of information between populations and increases the diversity of antibodies. In other words, polyclonal transforms the problem from a low dimensional space into a high dimensional space and then maps the result into the low dimensional space.

The antigen of immunity polyclonal algorithm corresponds to the objective function of optimal problem and various constraints, the antibody corresponds to the optimal solution and the affinity between antigen, and antibody corresponds to the matching degree of objective function and homographic solution. In this paper, the antigen corresponds to the local environmental information, the antibody corresponds to the individuals tending to the same extreme value in solution space of artificial immune network, and the affinity between antigen and antibody corresponds to m_i of artificial immune network.

Assuming the initial antibody population $\mathbf{A} = \{c_1, c_2, \ldots, c_n\}$, n is the size of initial antibody population, and c_i is the concentration of ith antibody which is one of individuals tending to the same extreme value in the solution space of artificial immune network. Polyclonal operator divided the point $c_i \in \mathbf{A}$ into q_i same points $c_i' \in \mathbf{A}'$, and then getting the new antibody population through clonal operator, clonal crossover operator, clonal mutation operator, and clonal selection operator. Polyclonal operator can be described as follows.

3.4.1. Clonal Operator. Defined as

$$\Theta(\mathbf{A}) = \left[\Theta(c_1), \Theta(c_2), \ldots, \Theta(c_n)\right]^T, \tag{15}$$

where $\Theta(c_i) = \mathbf{I}_i \times c_i$, $i = 1, 2, \ldots, n$ and $\mathbf{I}_i$ is a q_i-dimensional vector.

Generally

$$q_i = \mathrm{Int}\left(N \times \frac{m(c_i)}{\sum_{j=1}^{n} m(c_j)}\right), \quad i = 1, 2, \ldots, n. \tag{16}$$

$N > n$ is a given value related to the clonal scale and $\mathrm{Int}(x)$ rounds the argument x toward the least integer greater than x. $m(c_i)$ is the affinity between antigen and antibody, q_i is used to reflect the clonal scale of antibody. The previous equation implies that every antibody will be viewed locally and has its clonal scale different from the other ones. After cloning, the initial antibody population becomes the following population:

$$\mathbf{B} = \left\{\mathbf{A}, \mathbf{A}_1', \mathbf{A}_2', \ldots, \mathbf{A}_n'\right\}, \tag{17}$$

where

$$\mathbf{A}_i' = \left\{c_{i1}, c_{i2}, \ldots, c_{iq_i-1}\right\}, \tag{18}$$

where $c_{ij} = c_i$, $j = 1, 2, \ldots, q_i - 1$.

The range of clonal scale is [10, 100]. The bigger the clonal scale, the more time complexity of the algorithm. In this paper, the affinity between antigen and every antibody is same, so every antibody's clonal scale is same. Clonal scale of every antibody is $N_c = N \times B = 40$, B is clonal coefficient, and $0 < B < 1$, $B = 0.125$, $N = 320$.

3.4.2. Clonal Crossover Operator. Crossover selects two individuals from the population by the larger probability and exchanges one or some components of two individuals. Crossover guarantees the search of solution space will not trap into immature convergence because of high fitness individuals and makes the search stronger.

Clonal crossover operator maintains the information of initial population and does not act on $\mathbf{A} \in \mathbf{A}'$. The following crossover method is adopted. Considering two antibodies $c_i = \{x_1, x_2, \ldots, x_{N_c}\}$ and $c_j = \{y_1, y_2, \ldots, y_{N_c}\}$, the s_1th component is selected as crossover point; namely,

$$\begin{gathered} T_c^C\left(c_i, c_j\right) = \left\{x_1, \ldots, x_{s_1}, y_1, \ldots, y_{s_2}\right\}, \\ s_1 + s_2 = N_c, \quad c_i \in \mathbf{A}', \ c_j \in \mathbf{A}'. \end{gathered} \tag{19}$$

The range of crossover probability values are [0.5, 0.99]. In this paper, the crossover probability is $P_c = 0.8$.

3.4.3. Clonal Mutation Operator. Clonal mutation operator changes some genes in the group list of population, and the exchange is to put negation of some genetic value, that is, $0 \rightarrow 1$ or $1 \rightarrow 0$. The specific step is described as follows: the clonal mutation probability P_m and every gene of the path produce a random number $R \in [0, 1]$, if $P_m \geq R$, the gene produces mutation ($0 \rightarrow 1$ or $1 \rightarrow 0$).

The range of mutation probability values are [0.01, 0.05]. In this paper, the mutation probability is $P_m = 0.03$.

3.4.4. Clonal Selection Operator. For all $i = 1, 2, \ldots, n$, if there is a mutated antibody b such that $m(b) = \max\{m(c_i)\}$ satisfying the following

$$m(c_i) < m(b), \quad c_i \in \mathbf{A}', \tag{20}$$

then b replaces c_i in the antibody population.

In this paper, immunity polyclonal algorithm increases the diversity of antibodies which tend to the same extreme value and selects the antibody with highest concentration as the final antibody/steering direction. Thus, it overcomes the problems of immature convergence and local minima.

4. Simulation and Discussions

Some experiments are carried out for validating the proposed algorithm using MATLAB 2011 (a) GUI. The initial position of robot, goal, and obstacles get randomly in experimental

environment. Assume robot has eight uniformly distributed distance sensors (N_S = 8) and eight moving directions (N_{Ab} = 8) including forward, left, right, back, forward left, forward right, back left, and back right. Experimental environment is 20 × 20 m, obstacles are expressed with black square and the size of obstacles is 1 × 1 m.

4.1. PC-AIN Comparison with AIN

4.1.1. PC-AIN Improves Immature Convergence Problem of AIN. Artificial immune network (AIN) for mobile robot path planning exists as an immature convergence problem because some or all individuals in the solution space tend to the same extreme value. In this case, the robot will stop moving or move with wrong direction.

Experimental environment sets are as follows: the initial position of robot (9,8), the goal position (8,16), and the velocity of robot is 0.1 m/s. In Figures 2(a) and 2(b), artificial immune network is utilized for mobile robot path planning. There exists a condition that some or all individuals in the solution space of AIN tend to the same extreme value as shown in Figure 2(a). At this point robot randomly selects one of the directions to move because the moving probabilities toward several directions are equal. If this condition cannot improved effectively, robot will collide with obstacle as shown in Figure 2(b) and path planning fails at this time. The situation that some or all individuals in solution space tend to the same extreme value appeared, and robot's path is shown in Figure 2(b) expressed by black circle, and the selected antibody at every step when some or all concentrations in solution space tend to the same extreme value is shown in Figure 2(d) expressed by red "*".

Immunity polyclonal algorithm increases the diversity of antibodies which tend to the same extreme value in the solution space of AIN through clonal operator, clonal crossover operator, clonal mutation operator, and clonal selection operator. In polyclonal-based artificial immune network (PC-AIN) for mobile robot path planning, robot will move with the optimal direction at every step and get optimal path finally. Figure 2(c) shows polyclonal-based artificial immune network algorithm for mobile robot path planning under the experimental environment in Figure 2(a). The black circle represents the path when some or all individuals tend to the same extreme value in solution space of AIN, and the selected antibody at every step during this path is shown in Figure 2(d) expressed by black "*". From Figure 2(c), it can be seen that polyclonal-based artificial immune network (PC-AIN) solves the problem of immature convergence with increasing diversity of antibodies, and robot successfully reaches the goal in this condition.

Based on Figure 2, the selected antibody when some or all individuals tend to the same extreme value in solution space of AIN and PC-AIN is shown in Figure 2(d). It can be seen that artificial immune network (AIN) cannot effectively select the antibody with highest concentration, and robot traps into immature convergence problem when some or all individuals tend to the same extreme value in concentration solution space, while polyclonal-based artificial immune network (PC-AIN) successfully selects the antibody/steering direction with highest concentration and solves the problem of immature convergence with increasing the diversity of antibodies, and robot successfully reaches the goal.

Table 1: Performance comparison of AIN and PC-AIN.

Algorithm	PC-AIN	AIN
Running time(s)	6.37	8.13
Steps of path planning	157	227

4.1.2. Performance Comparison of AIN and PC-AIN. Experimental environment sets are as follows: the initial position of robot (5, 4), the goal position (10, 16), the number of obstacles are 5, the obstacles position (9, 10), (7, 6), (5, 8), (8, 13), and (6, 12), respectively. The velocity is 0.1 m/s. AIN and PC-AIN for mobile robot path planning is in the above environment. Table 1 is the results of performance comparison for mobile robot path planning between AIN and PC-AIN. The performance involves steps of path planning and running time from the initial position to the goal. From the property of running time, it can be seen that the running time of PC-AIN is less than the running time of AIN for the same operating environment. Meanwhile, from the comparison of steps, the length of PC-AIN is shorter than the length of AIN. AIN and PC-AIN for mobile robot path planning are shown in Figures 3(a) and 3(b), respectively.

From the above comparison, it can be seen that the polyclonal-based artificial immune network algorithm solves the problem of immature convergence with increasing the diversity of antibodies which tend to the same extreme value, and it has better optimal ability than artificial immune network.

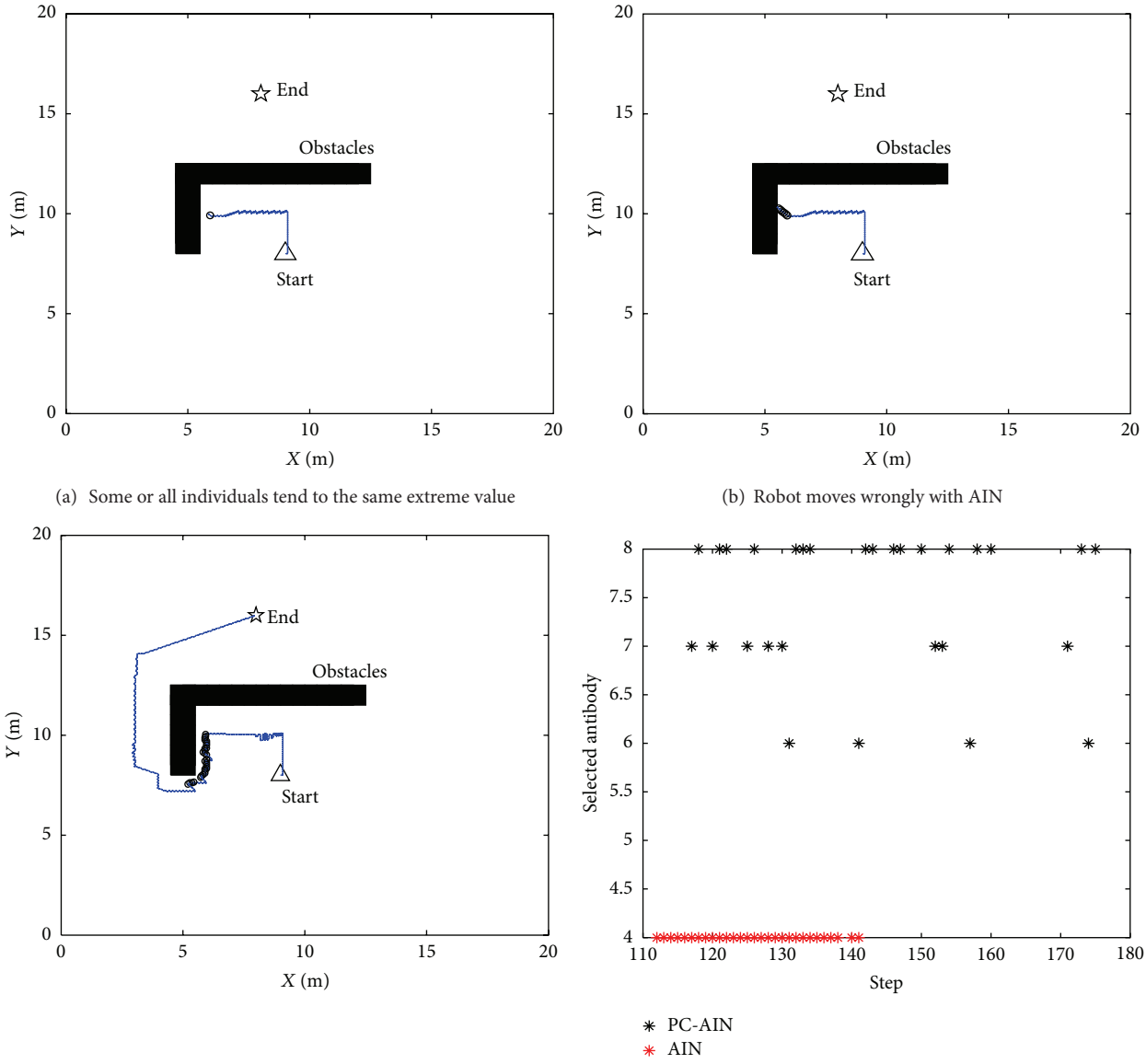
4.2. Local Minima.

A series of experiments are performed to compare the behavior of the proposed algorithm with reactive immune network [16] in solving the problem of local minima. Experimental environment sets are as follows: robot's initial position (3, 8), goal's initial position (12, 11), and obstacles' position (9, 10), (7, 11), (5, 8), and (12, 8). The velocity is 0.1 m/s and t = 1 s.

4.2.1. Reactive Immune Network. Virtual target method is used to solve local minima problem in reactive immune network (RIN). Robot moves to the goal under the interaction from virtual target, original goal, and obstacles. Revoking the virtual target until the robot is out of the local minima problem and then the robot moves to the original goal under the interaction from original goal and obstacles. When robot meets local minima again, it creates virtual target until reaching the goal.

In the above experimental environment, reactive immune network (RIN) is utilized for mobile robot path planning and the simulation result is shown in Figure 4(a).

4.2.2. Polyclonal-Based Artificial Immune Network. Polyclonal-based artificial immune network effectively



(a) Some or all individuals tend to the same extreme value

(b) Robot moves wrongly with AIN

(c) PC-AIN for path planning

(d) The selected antibody of AIN and PC-AIN at every step

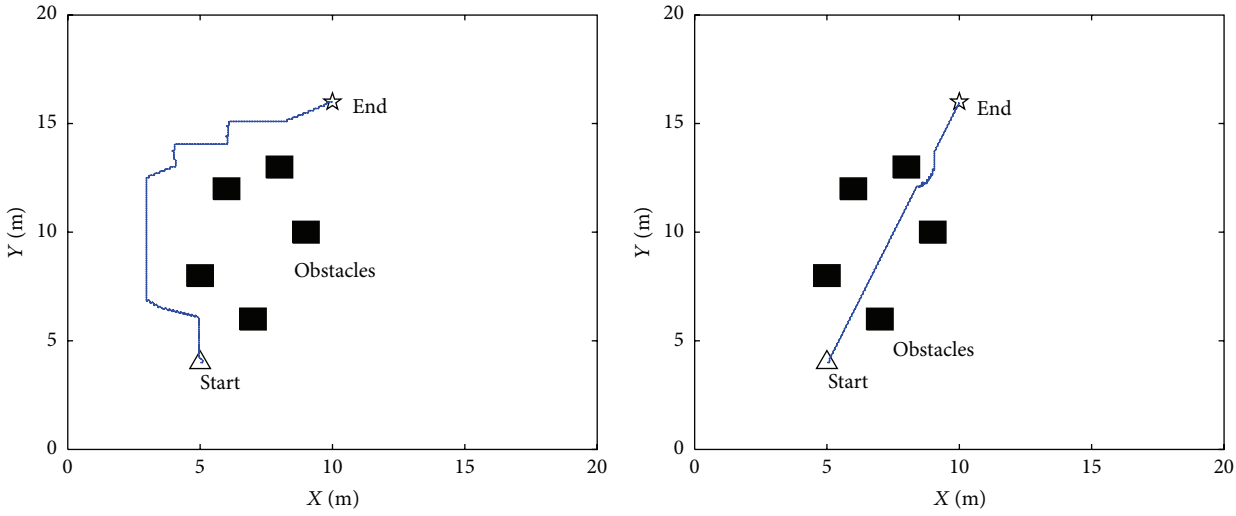
Figure 2: Mobile robot path planning with AIN and PC-AIN.

solves the problem of local minima caused by artificial potential field during the structure of parameter in artificial immune network through ploYclonal operator. Under the above environment, PC-AIN solves the problem of local minima, and the path planning is shown in Figure 4(b).

Figure 5 shows the comparison of steps from the initial position to the goal with reactive immune network and polyclonal-based artificial immune network. Figure 5 shows that path length of PC-AIN is less than the path length of RIN for mobile robot path planning, when the robot traps in local minima problem under the same experimental environment.

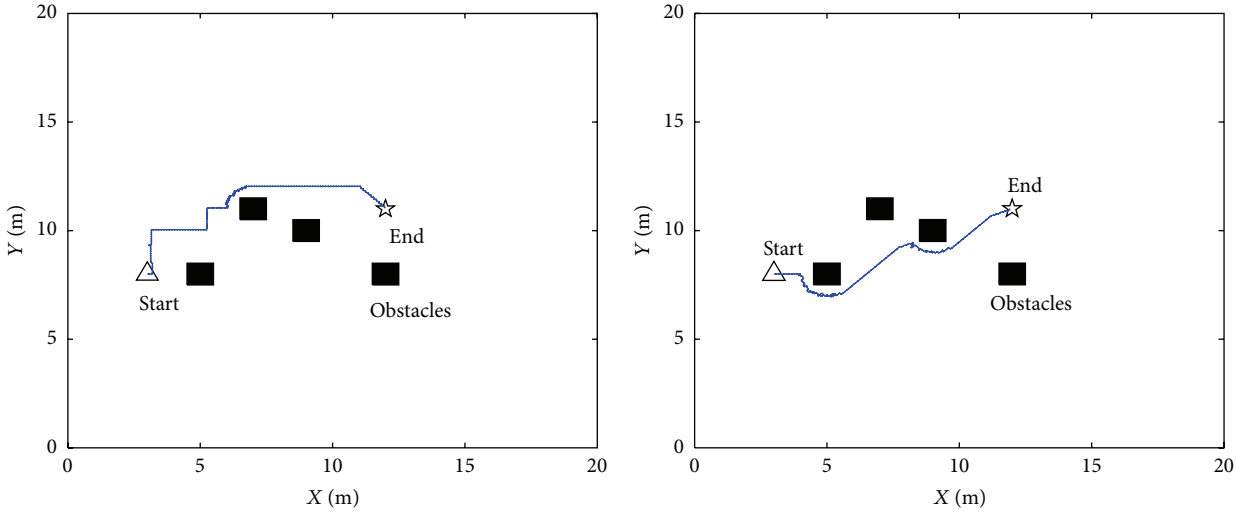
4.3. Robot's Moving Velocity Impact on the Convergence Rate. Robot's experimental environment is the same as Figure 3, where $t = 1$ s. When the robot moves at different velocity, the convergence rate of the algorithm is different. When the velocity within a certain range, robot's convergence rate increases with rising of the velocity. While when the velocity beyond a certain range, robot's convergence rate decreases with increasing of the velocity.

As shown in Figure 6, the robot moves at different velocities 0.05 m/s, 0.1 m/s, 0.15 m/s, 0.2 m/s, 0.25 m/s, and 0.3 m/s, respectively. With different velocities, robot has different running steps and different convergence rate. The result shows that when the velocity is less than 0.25 m/s, steps used reaching the goal are reduced and convergence rate is increased with rising of velocity. While when the robot's velocity is 0.25 m/s, steps of robot for path planning are greater than the steps with the speed of 0.2 m/s and 0.15 m/s, and when the robot's velocity is 0.3 m/s, steps of robot for path



(a) Path planning with AIN (b) Path planning with PC-AIN

Figure 3: Mobile robot path planning with AIN and PC-AIN.



(a) Path planning with RIN (b) Path planning with PC-AIN

Figure 4: Mobile robot path planning with RIN and PC-AIN.

planning are greater than the steps with the speed of 0.2 m/s. So, robot's convergence rate does not increase with rising of velocity.

Accordingly, the velocity of robot is not faster or better in mobile robot path planning, and the velocity has certain limitations. Selecting a proper velocity increases the convergence rate for mobile robot path planning.

4.4. The Interaction between Antibodies. This experiment is mainly used for validating stimulative affinity and suppressive affinity between antibodies, and expounding the antibody does not exist in the body independently. Figure 7 shows the affinity between antibody 1 and other antibodies.

There are eight antibodies in the experiment; affinity between antibody 1 and other antibodies expressed by red "*" in Figure 7. The other antibodies have stimulative affinity impact on antibody 1 if the affinity is greater than zero, and the other antibodies have suppressive affinity impact on antibody 1 if the affinity is less than zero. And the antibody 1 has the maximum stimulative affinity impact on antibody 1. The results correspond to (13). Stimulative-suppressive effect is positive if $0 < \Delta\theta_{il} < \pi/2$, or $3\pi/2 < \Delta\theta_{il} < 2\pi$, and stimulative-suppressive effect is negative if $\pi/2 < \Delta\theta_{il} < 3\pi/2$. The interaction between antibodies has a great impact on the immune network.

4.5. Analysis of Time Complexity. Polyclonal-based artificial immune network optimizes the path planning and speeds up the convergence rate. Meanwhile, it also solves the problems of immature convergence and local minima, but it's time

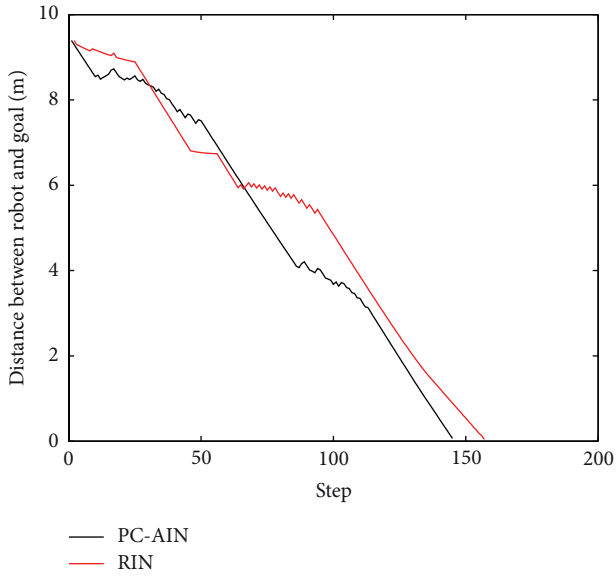


Figure 5: Path planning comparison of RIN and PC-AIN.

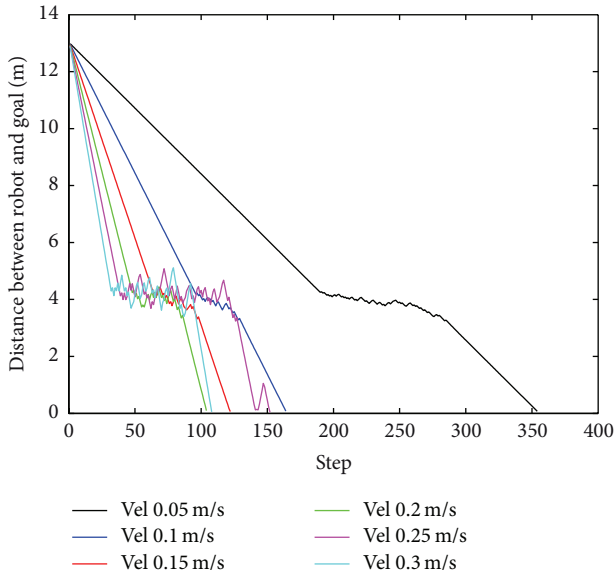


Figure 6: Moving velocity impact on convergence rate.

complexity of each generation increases because of the use of clonal operator, clonal crossover operator, clonal mutation operator, and clonal selection operator.

Assuming that n is the size of antibody population, $N > n$ is a given value related to the clonal scale and N_c is every antibody's clone scale. In this paper, $N_c = 5n$. At every generation of evolution, the worst time complexity of clonal operator is $O(nN_c)$, the worst time complexity of clonal crossover operator is $O(N)$, the worst time complexity of clonal mutation operator is $O(N)$, and the worst time complexity of clonal selection operator is $O(N + n)$. So,

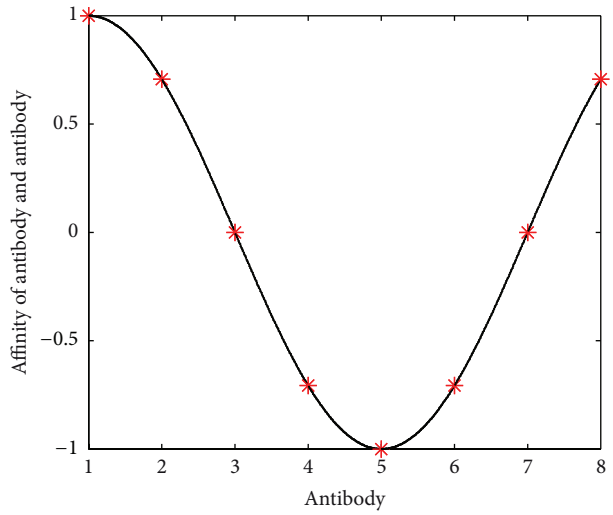


Figure 7: Affinity variation of antibody 1 with other antibodies.

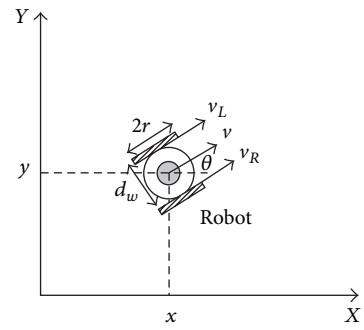


Figure 8: Differential drive mobile robot.

the worst time complexity of the proposed algorithm in the generation is computed as follows:

$$\begin{aligned} &O(nN_c) + O(N) + O(N) + O(N+n) \\ &\quad = O(nN_c + 3N + n) \\ &\quad = O(n \cdot 5n + 3n \cdot N_c + n) \\ &\quad = O(5n^2 + 3n \cdot 5n + n) \\ &\quad = O(5n^2 + 15n^2 + n) \\ &\quad = O(20n^2 + n). \end{aligned} \tag{21}$$

So, the worst time complexity of the proposed algorithm in the generation is $O(n^2)$.

4.6. Real-Time Simulation. In this experimental, amigo is used to validate the proposed algorithm, and programming languages used are MATLAB and C++. The system mainly consists of the following parts: host computer, vision system, amigo with an on-board PC, and wireless Ethernet. The camera is used to get the environmental information including obstacles and goal in this system. MATLAB is used to execute

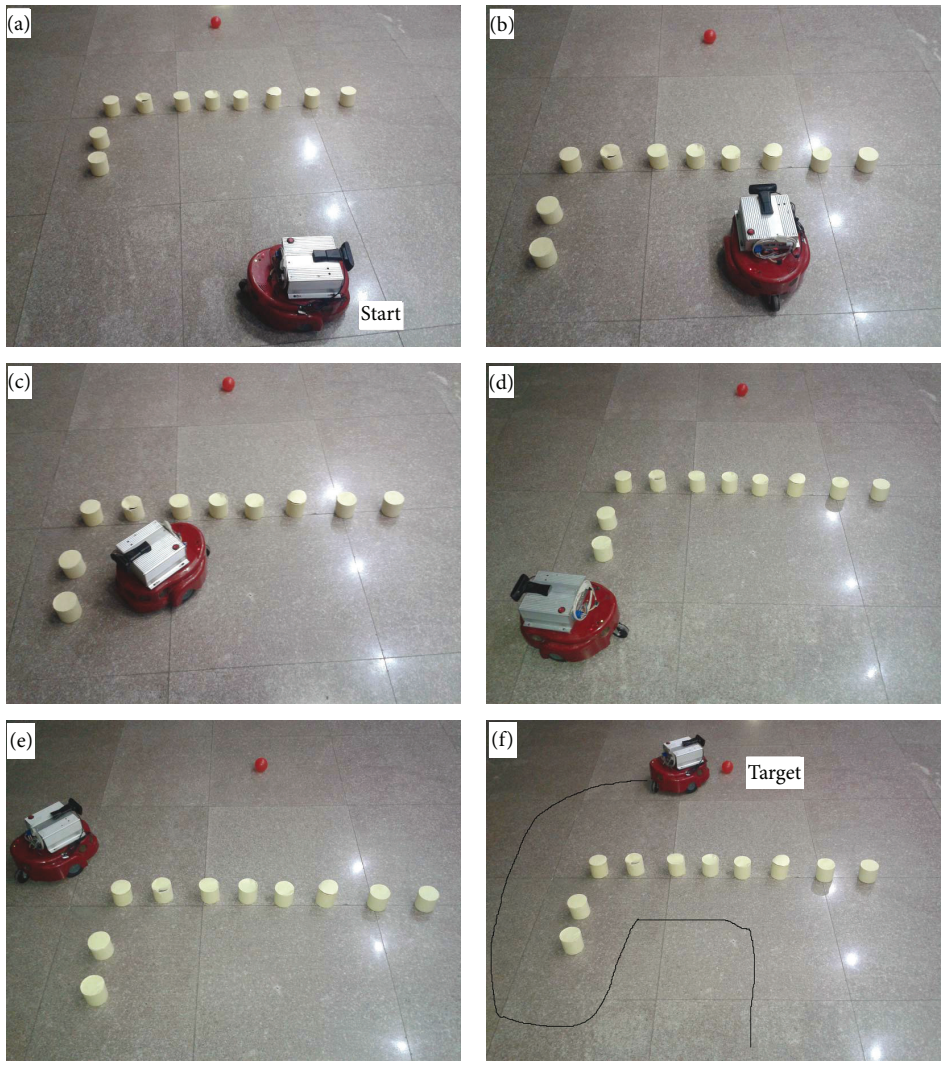


Figure 9: Barrier wall.

the path planning algorithm, and C++ is used to control the motion of robot. The host computer controls the on-board PC through wireless Ethernet.

The robot is a differential drive two-wheeled mobile robot. In Cartesian coordinates, the pose, velocity, and angular of differential drive two-wheeled mobile robot can be described as (22) [27]

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 \\ \sin\theta & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{22}$$

where v is the robot translation velocity and ω is the robot rotation velocity defined as (23)

$$\omega = \frac{v_R - v_L}{d_w}, \qquad v = \frac{v_R + v_L}{2}. \tag{23}$$

v_R and v_L are right and left wheels translation velocities of robot, respectively, and d_w is the distance between two wheels centers as shown in Figure 8.

In this paper, polyclonal-based artificial immune network is used to get the pose of robot. At real environment, two wheels velocities are computed through robot pose.

Assuming ΔS_R and ΔS_L are the displacement of right and left wheels during Δt, so

$$\begin{aligned} \Delta x &= \frac{\Delta S_L + \Delta S_R}{2} \times \cos\theta, \\ \Delta y &= \frac{\Delta S_L + \Delta S_R}{2} \times \sin\theta, \\ \Delta\theta &= \frac{\Delta S_R - \Delta S_L}{b}. \end{aligned} \tag{24}$$

Based on (24), the displacement of right and left wheels is computed as follows:

$$\begin{aligned} \Delta S_R &= \frac{\Delta x}{\cos\theta} + \frac{d_w \Delta\theta}{2}, \quad \cos\theta \neq 0, \\ \Delta S_L &= \frac{\Delta x}{\cos\theta} - \frac{d_w \Delta\theta}{2}, \quad \cos\theta \neq 0 \end{aligned} \tag{25}$$

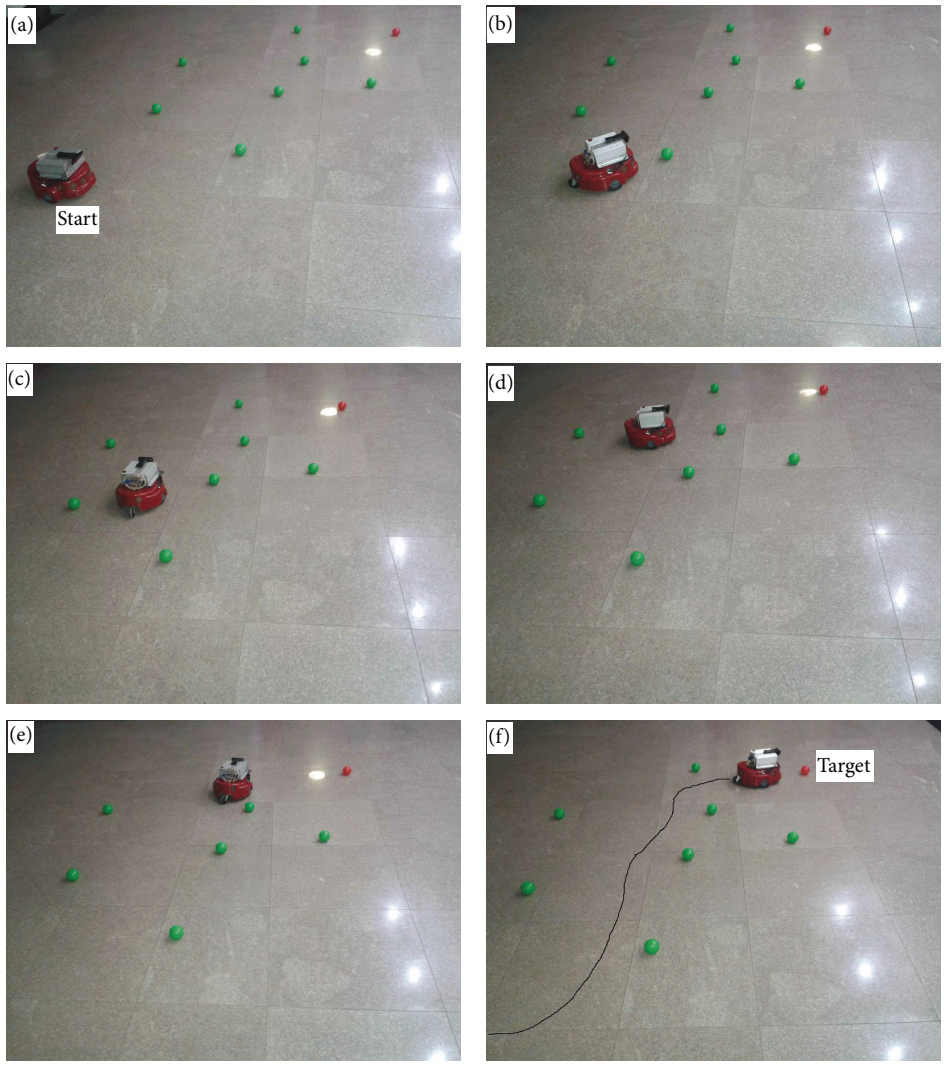


Figure 10: Complex environment.

or

$$\Delta S_R = \frac{\Delta y}{\sin\theta} + \frac{d_w \Delta\theta}{2}, \quad \sin\theta \neq 0,$$
$$\Delta S_L = \frac{\Delta y}{\sin\theta} - \frac{d_w \Delta\theta}{2}, \quad \sin\theta \neq 0. \tag{26}$$

From (25) and (26), left and right wheels velocities are computed.

The proposed algorithm is validated at the following scenarios as shown in Figures 9 and 10. Figure 9 is a barrier wall, and Figure 10 is a complex environment. Robot reaches the target successfully at the previous scenarios.

5. Conclusion

This paper deals with a polyclonal-based artificial immune network algorithm for mobile robot path planning in unknown static environment. This algorithm overcomes the immature convergence problem of artificial immune network and local minima problem of artificial potential field with increasing diversity of antibodies which tend to the same extreme value in solution space. Different simulated test scenarios are conducted to test the performance of the proposed algorithm. Simulation results validate the flexibility, efficiency, and effectiveness of the robot path planning architecture, especially the immature convergence and local minima problem. Meanwhile, some real-time experiments validate the implementability of the proposed algorithm.

Conflict of Interests

The authors declare that they have no conflict of interests.

Acknowledgments

This project was supported by the National Natural Science Foundation of China (nos. 61075091, 61105100, and 61240052), the Natural Science Foundation of Shandong

Province, China, (no. ZR2012FM036), and the Independent Innovation Foundation of Shandong University (nos. 2011JC011 and 2012JC005).

References

[1] X. Ma, Y. Xu, G.-Q. Sun, L.-X. Deng, and Y.-B. Li, "State-chain sequential feedback reinforcement learning for path planning of autonomous mobile robots," *Journal of Zhejiang University Science C*, vol. 14, no. 3, pp. 167–178, 2013.

[2] R. Abiyev, D. Ibrahim, and B. Erin, "Navigation of mobile robots in the presence of obstacles," *Advances in Engineering Software*, vol. 41, no. 10-11, pp. 1179–1186, 2010.

[3] Q. Zhang, D. Chen, and T. Chen, "An obstacle avoidance method of soccer robot based on evolutionary artificial potential field," *Journal of Energy Procedia*, vol. 16, pp. 1792–1798, 2012.

[4] R. A. F. Romero, E. Prestes, M. A. P. Idiart, and G. Faria, "Locally oriented potential field for controlling multi-robots," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4664–4671, 2012.

[5] M. A. Kareem Jaradat, M. Al-Rousan, and L. Quadan, "Reinforcement based mobile robot navigation in dynamic environment," *Robotics and Computer-Integrated Manufacturing*, vol. 27, no. 1, pp. 135–149, 2011.

[6] N. Navarro-Guerrero, C. Weber, P. Schroeter, and S. Wermter, "Real-world reinforcement learning for autonomous humanoid robot docking," *Journal of Robotics and Autonomous Systems*, vol. 60, pp. 1400–1407, 2012.

[7] O. Motlagh, S. H. Tang, N. Ismail, and A. R. Ramli, "An expert fuzzy cognitive map for reactive navigation of mobile robots," *Journal of Fuzzy Sets and Systems*, vol. 201, pp. 105–121, 2012.

[8] M. T. Ibrahim, D. Hanafi, and R. Ghoni, "Autonomous navigation for a dynamical hexapod robot using fuzzy logic controller," *Journal of Procedia Engineering*, vol. 38, pp. 330–341, 2012.

[9] O. Castillo, R. Martínez-Marroquín, P. Melin, F. Valdez, and J. Soria, "Comparative study of bio-inspired algorithms applied to the optimization of type-1 and type-2 fuzzy controllers for an autonomous mobile robot," *Journal of Information Sciences*, vol. 192, pp. 19–38, 2012.

[10] A. Tuncer and M. Yildirim, "Dynamic path planning of mobile robots with improved genetic algorithm," *Journal of Computers and Electrical Engineering*, vol. 38, pp. 1564–1572, 2012.

[11] M. Aly and A. Abbas, "Simulation of obstacles effect on industrial robots working space using genetic algorithm," *Journal of King Saud University—Engineering Sciences*, 2013.

[12] B. Deepak, D. R. Parhi, and S. Kundu, "Innate immune based path planner of an autonomous mobile robot," *Journal of Procedia Engineering*, vol. 38, pp. 2663–2671, 2012.

[13] X. Chen, G.-Z. Tan, and B. Jiang, "Real-time optimal path planning for mobile robots based on immune genetic algorithm," *Journal of Central South University (Science and Technology)*, vol. 39, no. 3, pp. 577–583, 2008.

[14] A. Raza and B. R. Fernandez, "Immuno-inspired heterogeneous mobile robotic systems," in *Proceedings of the 49th IEEE Conference on Decision and Control (CDC '10)*, pp. 7178–7183, December 2010.

[15] M. Yuan, S.-A. Wang, C. Wu, and N. Chen, "A novel immune network strategy for robot path planning in complicated environments," *Journal of Intelligent & Robotic Systems*, vol. 60, no. 1, pp. 111–131, 2010.

[16] G.-C. Luh and W.-W. Liu, "An immunological approach to mobile robot reactive navigation," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 30–45, 2008.

[17] R. Challoo, P. Rao, S. Ozcelik, L. Challoo, and S. Li, "Navigation control and path mapping of a mobile robot using artificial immune systems," *Journal of Robotics and Automation*, vol. 1, no. 1, pp. 1–25, 2010.

[18] S. Ozcelik and S. Sukumaran, "Implementation of an artificial immune system on a mobile robot," *Journal of Procedia Computer Science*, vol. 6, pp. 317–322, 2011.

[19] T. Bonaci, P. Lee, L. Bushnell, and R. Poovendran, "A convex optimization approach for clone detection in wireless sensor networks," *Journal of Pervasive and Mobile Computing*, vol. 9, no. 4, pp. 528–545, 2013.

[20] L. N. De Castro and F. J. Von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.

[21] J. Chen and M. Mahfouf, "A population adaptive based immune algorithm for solving multi-objective optimization problems," in *Proceedings of the International Symposium on Artificial Immune Systems*, pp. 280–293, Springer, 2006.

[22] N. C. Cortes and C. A. C. Coello, "Multiobjective optimization using ideas from the clonal selection principle," in *Proceedings of the International Symposium on Genetic and Evolutionary Computation*, pp. 158–170, 2003.

[23] G. Meng and C. Qiuhong, "A new immune algorithm and its application," *WSEAS Transactions on Computers*, vol. 9, no. 1, pp. 72–82, 2010.

[24] C. Huizar, O. Montiel-Ross, R. Sepulveda, and F. J. D. Delgadillo, "Path planning using clonal selection algorithm," in *Proceedings of the International Symposium on Hybrid Intelligent Systems*, pp. 303–312, Springer, 2013.

[25] R.-C. Liu, H.-F. Du, and L.-C. Jiao, "An immune monoclonal strategy algorithm," *Acta Electronica Sinica*, vol. 32, no. 11, pp. 1880–1884, 2004.

[26] Y. Shen and M. Yuan, "A novel poly-clone particle swarm optimization algorithm and its application in mobile robot path planning," in *Proceedings of the Chinese Control and Decision Conference (CCDC '10)*, pp. 2271–2276, May 2010.

[27] S.-W. Yu and W.-J. Yan, "Design of low-level motion controller for a two-wheel mobile mini-robot," *Journal of Mechanical and Electrical Engineering Magazine*, vol. 23, no. 9, pp. 38–46, 2006.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 821315, 6 pages
http://dx.doi.org/10.1155/2013/821315

Research Article

Real-Time Detection of Application-Layer DDoS Attack Using Time Series Analysis

Tongguang Ni, Xiaoqing Gu, Hongyuan Wang, and Yu Li

School of Information Science and Engineering, Changzhou University, Changzhou 213164, China

Correspondence should be addressed to Hongyuan Wang; tiddyddd@163.com

Received 7 June 2013; Accepted 25 August 2013

Academic Editor: Xiaomei Qi

Distributed denial of service (DDoS) attacks are one of the major threats to the current Internet, and application-layer DDoS attacks utilizing legitimate HTTP requests to overwhelm victim resources are more undetectable. Consequently, neither intrusion detection systems (IDS) nor victim server can detect malicious packets. In this paper, a novel approach to detect application-layer DDoS attack is proposed based on entropy of HTTP GET requests per source IP address (HRPI). By approximating the adaptive autoregressive (AAR) model, the HRPI time series is transformed into a multidimensional vector series. Then, a trained support vector machine (SVM) classifier is applied to identify the attacks. The experiments with several databases are performed and results show that this approach can detect application-layer DDoS attacks effectively.

1. Introduction

DDoS attacks have caused severe damage to servers and will cause even greater intimidation to the development of new Internet services. DDoS attacks are categorized into two classes: network-layer DDoS attacks and application-layer DDoS attacks. In network-layer DDoS attacks, attackers send a large number of bogus packets towards the victim server and normally attackers use IP spoofing. The victim server or IDS can easily distinguish legitimate packets from DDoS packets. In contrast, in application-layer DDoS attacks, attackers attack the victim server through a flood of legitimate requests. In this attack model, attackers attack the victim Web servers by HTTP GET requests and pulling large files from the victim server in overwhelming numbers. Also, attackers can run a massive number of queries through the victim's search engine or database query to bring the server down.

To circumvent detection, the attackers increasingly move away from pure bandwidth floods to stealthy DDoS attacks that masquerade as flash crowd. Flash crowd [1, 2] refers to the situation when a very large number of users simultaneously access a website, which may be due to the announcement of a new service or free software download. Because burst traffic and high volume are the common characteristics of application-layer DDoS attacks and flash crowd, it is not easy to distinguish them. Therefore, application layer DDoS attacks may be stealthier and more dangerous for the websites than the general network-layer DDoS attacks.

Most well-known DDoS countermeasure [3] techniques are against network-layer DDoS attacks. Those techniques cannot handle application-layer DDoS attacks. Countering application-layer DDoS attacks becomes a great challenge. Statistical methods is used to detect characteristics of HTTP sessions and employed rate-limiting as the primary defense mechanism in [4]. Constraint random request attacks by the statistical methods are used to defend against the application-layer DDoS attacks in [5]. A CAPTCHA puzzle is used to ensure that the response is generated by a human not by a machine in [6]. A semi-Markov model is proposed to describe the browsing Behaviors of Web surfers in [7], and an improved semi-Markov model is proposed to describe the dynamic behavior process of aggregated traffic in [8]. Recently, trust-based methods [9, 10] were introduced for resisting application-layer DDoS attacks. The common feature of these methods is that a defense system establishes credit records for each user. The credit value given to a sender is designed to be measured based on its history of communication patterns.

In application-layer DDoS attacks, attack sources have been programmed and worked according to their attack

functions, so detection based on its pattern is possible. In this paper, the entropy of HTTP GET requests per source IP address (HRPI) is proposed, which reflects the essential features of application-layer DDoS attacks: the distribution of source IP address and HTTP GET request frequency. To increase the detection accuracy in various conditions, HRPI time series are transformed into a multidimensional vector by estimating the adaptive autoregressive (AAR) model parameters using Kalman filter. Furthermore, a support vector machine (SVM) classifier, which is trained by AAR parameters of HRPI time series, is applied to classify the state of current network traffic and identify the application-layer DDoS attacks.

The rest of the paper is organized as follows. Section 2 discusses the application-layer DDoS attacks and details their properties. Section 3 describes our approach to detect the application-layer DDoS attacks. In Section 4, experiments are presented to validate our detection model. Finally, the conclusion is given in Section 5 and it points out the future work.

2. Application-Layer DDoS Attacks

Application-layer DDoS attacks can be clustered into two types: bandwidth exhausting (HTTP flooding) and resources exhausting [11]. In bandwidth exhausting DDoS attacks, attackers attack the victim server through a flood of legitimate requests. Any zombie machine has to establish a TCP connection with the victim server, which requires a genuine IP address. Attacks mainly focus on the homepage or a hot webpage, and also different web pages. In this case, the sources of the traffic converge to a group of points and high HTTP Get request rate from the attackers.

Besides the flooding attack pattern, application-layer DDoS attacks may focus on exhausting the server resources such as Sockets, CPU, memory, disk/database bandwidth, and I/O bandwidth. With increasing computational complexity in Internet applications and larger network bandwidth, server resources may become the bottleneck of these applications. This type of attack is able to use fewer zombies but the attack has an even larger damage to the website. However, the traffic will be similar to the bandwidth exhausting DDoS. As a result, the sources of the traffic converge to a group of points but the targets of the traffic become dispersed in some extent. At the same time, the frequency of HTTP Get request from the attackers is highly large.

On the Web, flash crowd refers to the situation when a very large number of users simultaneously access a popular website, which produces a surge in traffic to the website and might cause the site to be virtually unreachable. DDoS attacks are absolutely different from flash crowd, DDoS attacks are due to an increase in the request rates for a small group of clients while flash crowd is due to an increase in the number of clients. The sources of flash crowd are definitely scattered, conversely, the sources of application-layer DDoS attacks converge to a group of points.

3. Our Approach

3.1. Definition of HRPI. For popular websites, the traffic targeted is a stream of successive HTTP Get requests.

Definition 1. HTTP Get requests in the certain time interval Δt is given in the form of $\langle (x_1, s_1), (x_2, s_2), \ldots, (x_n, s_n) \rangle$. For the (x_i, s_i), x_i is the source IP address and s_i is the number of HTTP Get requests for x_i.

Definition 2. Entropy of HTTP GET requests per source IP address (HRPI) is defined as

$$\mathrm{SRE} = -p(x_i) \sum lbp(x_i), \tag{1}$$

where $p(x_i)$ is the probability of HTTP Get requests belonging to x_i, and $p(x_i) = s_i / \sum_{i=1}^{n} s_i$.

HRPI as a summarization tool is used to quantify the degree of dispersal or concentration of HTTP Get request feature distributions. According to the analysis in Section 2, we deduced the following conclusion (DDoS as 1, normal as 2, flash crowd as 3):

$$\mathrm{HRPI}(3) > \mathrm{HRPI}(2) > \mathrm{HRPI}(1). \tag{2}$$

In most cases, distribution form of source IP address of legitimate users is more uniformly scattered across the Internet; the distribution form of source IP address of attackers is more cumulative in someplaces. In DDoS attacks, several clusters of source IP addresses and lager number of HTTP GET requests are converged, so HRPI value dramatically drops when attacks happen. Conversely, the sources of flash crowd are scattered and there were no such clusters, so it will result in an abnormal increase in HRPI of the network.

3.2. Generation of HRPI Time Series. Adaptive autoregressive AAR (p) model [12] of degree p is defined as

$$y_t = \sum_{k=1}^{p} a_t^k y_{t-k} + e_t, \tag{3}$$

where y_t denotes the observation at instant t and a_t^k denotes the time-varying model parameters. As the traffic collecting device may cause measurement errors, stochastic variable e_t is used to capture this error. The model uses a weighted sum of p previous values to estimate the current observation value. The weights a_t^k ($k = 1, \ldots, p$) are time dependent, and the current value can be predicted as a linear combination of p past values. By using time-varying AAR model, we allow a model of normal behavior to adapt to the changes of the monitored system.

Kalman filter is an adaptive and recursive data processing algorithm that is suited for online estimation [13, 14]. Kalman filter can process traffic matrix as a whole and all traffic can be estimated simultaneously. This implies that we do not have to consider all the previous data again, to compute the optimal estimates; we only need to consider the estimates from the previous time step and the new measurement.

In our case, we estimate the AAR model parameters from the observed alert series $\{y_t\}$. The true parameters cannot be observed directly and in state space terminology they are called the state X. Now, assume that we have an observation model giving the relation between the unobservable state

and the observations, and an evolution model describing the time-varying nature of the state. So the AAR model can be put in vector form as follows:

$$Y_t = H_t X_t + e_t, \tag{4}$$

where H_t denotes an internal matrix and $H_t = (Y_t, \ldots, Y_{t-p})$. X_t denotes the state vector at instant t and $X_t = (a_t^1, \ldots, a_t^p)^T$.

Without prior information, the evolution of the state is often described with a random walk model [15]. A linear equation is constructed as follows to build a prediction model to correlate X_{t+1} and X_t:

$$X_{t+1} = X_t + w_t, \tag{5}$$

where state noise $w_t$and measurement noise e_t are uncorrelated, zero-mean white-noise processes and with covariance matrices σ_w^2 and σ_e^2, respectively.

For representing the Kalman filter equations, $\widehat{X}$ denotes the estimate of X and $P_{t|t-1}$ denotes error covariance matrix for estimation error of the state at instant t using observations accumulated at instant $t-1$. When initial conditions, $\widehat{X}_0 = E[X_0]$ and error covariance matrix $P_0 = E[(\widehat{X}_0 - X_0)(\widehat{X}_0 - X_0)^T]$, the system state $\widehat{X}_{t|t-1}$ can be estimated iteratively by the following equations:

$$\begin{aligned}
\widehat{X}_{t|t-1} &= X_{t-1}, \\
P_{t|t-1} &= P_{t-1} + C_{w_{t-1}}, \\
K_t &= P_{t|t-1} H_t^T \left(H_t P_{t|t-1} H_t^T + C_{e_t}\right)^{-1}, \\
\widehat{X}_t &= \widehat{X}_{t|t-1} + K_t \left(Y_t - H_t \widehat{X}_{t|t-1}\right), \\
P_t &= \left(I - K_t H_t\right) P_{t|t-1}.
\end{aligned} \tag{6}$$

Using Kalman filter in practice requires initial values for state X_0, error covariance P_0, state noise covariance C_w ($C_w = \sigma_w^2 I$), and observation noise covariance C_e ($C_e = \sigma_e^2 = I$). A common approach is to set $X_0 = 0$, $P_0 = I$, and run the algorithm on a short segment from observation data backwards. The values obtained in this way for X and P_0 are then used to initialize these values in the actual processing run. The adaptation speed increases with C_w and the variance of state estimates is inversely proportional to the value of C_w. Therefore, it should be chosen for a desired balance between state estimate variance and filter adaptation speed according to the application. C_w needs to be set dynamically and according in application.

3.3. Kalman Filter Smoothing. There are three classical smoothing algorithms, fixed-point smoother, fixed-interval smoother, and fixed-lag smoother. We use fixed-lag smoother, since it is suitable for online processing when a small, fixed delay of L observations is allowed [16].

To estimate the state X_t at instant t with a fixed-lag smoother, we will wait to have observations up to instant $t+L$, where $L > 0$. The state and observation equations have now extended variables. The Kalman filter equations remain the same, and the observation (4) becomes

$$Y_t = [H_t, 0, \ldots, 0] \begin{bmatrix} X_t \\ X_{t-1} \\ \vdots \\ X_{t-L} \end{bmatrix} + e_t. \tag{7}$$

The simplified state (5) can be written as

$$\begin{bmatrix} X_{t+1} \\ X_t \\ X_{t-1} \\ \vdots \\ X_{t-(L-1)} \end{bmatrix} = \begin{bmatrix} X_t \\ X_{t-1} \\ X_{t-2} \\ \vdots \\ X_{t-L} \end{bmatrix} + \begin{bmatrix} w_t \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{8}$$

3.4. SVM Classifier. By sampling the network traffic with time interval Δt, calculating the HRPI of every sample, the HRPI sample series $\{\text{HRPI}_i, i = 1, 2, \ldots, N\}$ is gotten, N is the length of the series. Based on (6)–(8), multidimensional vector $\widehat{X}(i)$ of degree p can be used to describe the state features of network traffic. As a result, detecting DDoS attacks equates to classifying $\widehat{X}(i)$ series virtually.

Support vector machine (SVM) is applied here, which is a well-known data classification technique, to classify AAR parameters vector. SVM method can get the optimal solution whether the sample size tends to be finite or infinite. It can establish a mapping of a nonlinear kernel function, structuring the optimal hyperplane, so problem can be converted into a linearly separable one in the high-dimensional feature space. Besides, it solves the dimension problem and its complexity has nothing to do with the sample's dimension.

Since traffic is only considered as legitimate or attack, it is naturally a binary classification problem. The SVM classifier can be described as

$$\eta = \sum_{i=1}^{M} \alpha_i y_i K(\varphi_i, \varphi) + b, \tag{9}$$

where η is the classification result for the sample, α_i is the Lagrange multiplies, y_i is the category, and $y_i \in \{-1, 1\}$. $K(\varphi_i, \varphi)$ is the kernel function and b is the deviation factor.

The optimal hyperplane that SVM classifier created in the high-dimensional feature space is

$$f(\varphi) = \operatorname{sgn}\left(\sum_{i \in \text{SV}} \alpha_i y_i \left(K(\varphi_r, \varphi_i) + K(\varphi_s, \varphi_i)\right)\right), \tag{10}$$

where

$$b = \frac{1}{2} \sum_{i \in \text{SV}} \alpha_i y_i \left(K(\varphi_r, \varphi_i) + K(\varphi_s, \varphi_i)\right). \tag{11}$$

SV (Support Vector) denotes the support vector and φ_r means positive support vector, φ_s means negative support vector.

The coefficient can be obtained by the following quadratic programming:

$$\begin{aligned} \max \quad & w(a) = \sum_{i=1}^{n} a_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} a_i a_j y_i y_j K\left(x_i, x_j\right) \\ \text{s.t.} \quad & \sum_{i=1}^{n} a_i y_i = 0 \\ & 0 \le \alpha_i \le C \quad (i = 1, 2, \ldots, M), \end{aligned} \tag{12}$$

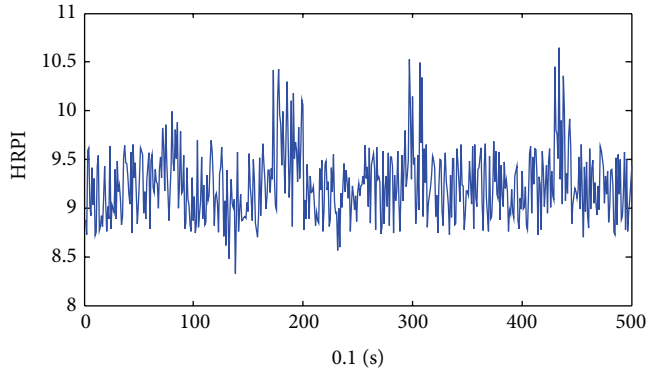
where C is the parameter to price the misclassification. Before the SVM can classify traffics, it should undergo a training process to develop a classification model. We use the LibSVM library [17] to implement SVM.

4. Experiments

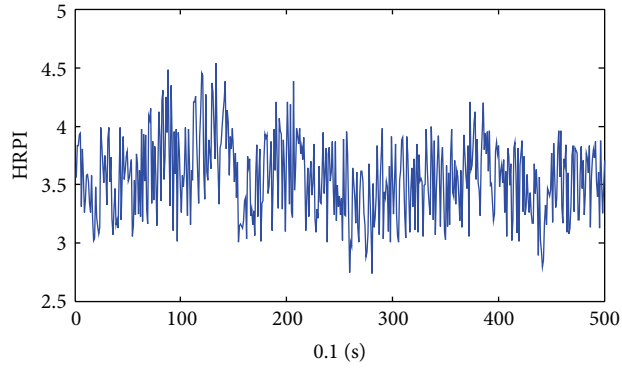
In order to evaluate the performance of our scheme, we divided our study into two groups of experiments: to detect application layer DDoS in normal traffic and in flash crowd.

4.1. Dataset. Normal traffic is the real-life Internet traces collected from the traffic archive of Changzhou university WWW server. The traces contain two weeks worth of all HTTP requests to the web server. We implemented application-layer DDoS attack in a simulator. Simulations are carried out using NS-2 network simulator on Linux platform. For generating attack traffic, there are 50 zombie machines and a web server. Attack rates are 20 HTTP Get requests/s, 30 HTTP Get requests/s, . . . , 60 HTTP Get requests/s, which simulate the attack rates of worm "Mydoom", and every attack lasts 1800s. Flash crowd is collected from the World Cup 98 website [18]. As this is a high arrival rate, we expect our approach to detect this traffic as flash crowd.

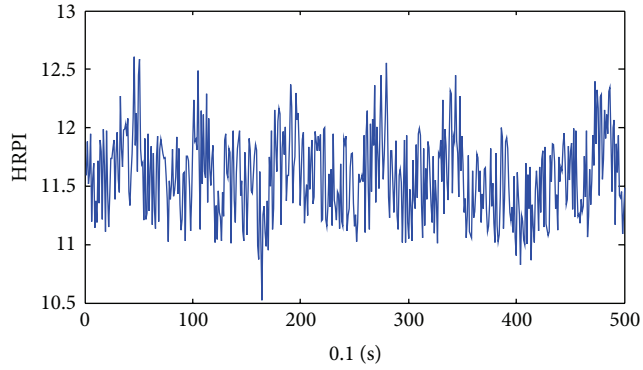
We obtained HRPI time series by multiple sampling and calculation when the sampling interval Δt is 0.1 s. As shown in Figure 1(a), HRPI of normal traffic varies with the time and its mathematical expectation is 9.26. Figure 1(b) shows HRPI of DDoS attack and its mathematical expectation is 3.58, and HRPI of flash crowd is shown in Figure 1(c) with mathematical expectation 11.57. We can see that the HRPI time series are sensitive to DDoS attack and flash crowd, so HRPI can distinguish three types of traffic distinctly.



(a) HRPI of normal traffic



(b) HRPI of DDoS attack



(c) HRPI of flash crowd

Figure 1: HRPI of different web traffics.

4.2. Model Parameters. There are three parameters which may affect the HRPI time series performances. The first one is the parameter p of AAR model. In practice, the model degree is often fixed using some prior knowledge or guidelines. To optimize the goodness of fit verse, model complexity ratio, and also to ease the computational load, we settled $p = 3$ as a degree which allowed the model to capture sufficiently well the normal traffic behavior.

The second one is state noise covariance C_w ($C_w = \sigma_w^2 I$). The adaptation speed of the Kalman filter is determined by the state noise covariance factor σ_w^2. It controls how fast the state adopts the changes in observations and gives a suitable balance in adapting to normal behavior and avoiding incorporating anomalous behavior in to the model. We experimented with different values and chose to use $\sigma_w^2 = 0.0001$.

The third one is the lag of Kalman filter, and we noticed a significant increase in model accuracy when $L = 1$. The increase in accuracy was slower when further increasing L. As larger L means also longer delay in detection, we chose to use $L = 1$.

4.3. Experiments and Results

4.3.1. Evaluation Criteria. In this paper, a group of performance metrics in classification problems are used for the evaluation of the results, consisting of FPR, FNR, accuracy,

Table 1: The results of DDoS detection in normal traffic.

	Accuracy	FPR	FNR	Precision	Recall	ROC
P-20	91.52%	8.24%	7.27%	91.62%	92.51%	90.12%
P-30	94.26%	5.05%	3.94%	94.51%	95.28%	95.31%
P-40	96.31%	3.74%	3.08%	96.79%	97.00%	98.79%
P-50	97.24%	2.76%	2.23%	97.30%	97.64%	99.10%
P-60	97.81%	2.05%	1.88%	97.94%	98.02%	99.26%

Table 2: The results of DDoS detection in flash crowd.

	Accuracy	FPR	FNR	Precision	Recall	ROC
T-20	92.33%	6.69%	6.03%	92.66%	93.28%	91.67%
T-30	95.02%	4.10%	3.09%	95.64%	96.37%	96.34%
T-40	96.39%	3.52%	2.58%	96.82%	96.95%	98.99%
T-50	97.68%	2.16%	1.94%	97.22%	98.03%	99.61%
T-60	98.06%	1.47%	1.02%	98.29%	98.49%	99.70%

precision, recall, and ROC. Let TP represent the normal test samples that have been correctly classified and let FP represent the ones that have been wrongly classified. Let TN represent the attacking test samples that have been correctly classified and let FN represent the ones that have been falsely classified. Thus, the False-Positive Rate (FPR) and the False-Negative Rate (FNR) are the proportions of wrongly classified normal test samples and attacking test samples, respectively (FPR = FP/(FP + TN), FNR = FN/(TP + FN)). Accuracy states the overall percentage of correctly classified attacking test samples (accuracy = (TP + TN)/(TP + FP + TN + FN)). Precision as the classifier's safety, states the degree in which messages identified as attacking test samples are indeed malicious (precision = TP/(TP + FP)). Recall as the classifier's effectiveness, states the percentage of attacking test samples that the classifier manages to classify correctly (recall = TP/(TP + FN)). Receiver Operating Characteristic (ROC) as a classifier's balance ability between its FPR and its FNR is a function of varying classification threshold.

4.3.2. Experiment 1: Detect DDoS Attacks in Normal Traffic. We set that the sampling interval Δt is 0.1 s, HRPI time series length N is 100, so the detection time is 10 s. In this experiment, normal traffic contain 600 series, and DDoS attack traffic contain 450 series. Obtained dataset is divided into two parts: training data contains 60% of total data values, testing data contains the rest of the obtained dataset. The kernel function in SVM classifier is radial basis function (RBF) and the robustness of the classifiers is evaluated using 10-fold cross-validation. In order to test the robustness of our method to the disturbance of normal traffic, we do five experiments named as P-20, P-30,. . ., P-60, in which traffic attacks are 20 HTTP Get requests/s,. . ., 60 HTTP Get requests/s mixing normal traffic at the same time.

Table 1 shows the performance results; the detection ratio of our approach increases when the attack traffic volume increases. When normal traffic is much larger than attack traffic, the detection ratio still keeps a high level. This means that our approach can identify the DDoS attack traffic with a high precision, and be sensitive to DDoS attack traffic.

4.3.3. Experiment 2: Detect DDoS Attacks in Flash Crowd. In this experiment, the sampling interval Δt is 0.1 s, and HRPI time series length N is 100, too. We sample flash crowd 500 series, and DDoS attack traffic 350 series. The training and testing method of SVM is the same as experiment 1. We do five experiments named as T-20, T-30, . . . , T-60, by mixing attacking traffic 20 HTTP Get requests/s, . . . , 60 HTTP Get requests/s and flash crowd.

Table 2 shows the performance results, with the increment of flash crowd, the detection ratio of our approach does not decline rapidly. The FPR and FNR are reduced with the increase of attack rate, and the accuracy, precision, recall, and ROC are ascended with the increase of attack rate.

In the above two groups of experiments, the false negatives come mainly from two aspects: firstly, due to the increase of normal traffic or flash crowd, which makes the HRPI states learn to normal ones, thus making the difference too small for detection. Secondly, the network state shift caused by network random noise results in false negative.

5. Conclusion

Application-layer DDoS attacks detection is a hot and difficult research topic in the field of intrusion detection. Based on the characteristics of DDoS attack, this paper proposes a novel approach to detect DDoS attacks. The work provides two contributions: (1) HRPI is introduced to detect DDoS attacks, and it reflects the essential features of attacks and (2) a detection scheme against DDoS attacks is proposed, and it can achieve high detection efficiency and flexibility.

In our future work, we will make a detailed study of how to set all kinds of parameters in different application scenarios adaptively.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Contact (61070121).

References

[1] T. Thapngam, S. Yu, W. Zhou, and G. Beliakov, "Discriminating DDoS attack traffic from flash crowd through packet arrival patterns," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM '11)*, pp. 952–957, April 2011.

[2] G. Oikonomou and J. Mirkovic, "Modeling human behavior for defense against flash-crowd attacks," in *Proceedings of the IEEE International Conference on Communications (ICC '09)*, pp. 1–6, June 2009.

[3] H. Beitollahi and G. Deconinck, "Analyzing well-known countermeasures against distributed denial of service attacks," *Computer Communications*, vol. 35, pp. 1312–1332, 2012.

[4] S. Ranjan, R. Swaminathan, M. Uysal, and E. Knightly, "DDoS-resilient scheduling to counter application layer attacks under

imperfect detection," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, pp. 1–13, April 2006.

[5] W. Yen and M.-F. Lee, "Defending application DDoS with constraint random request attacks," in *Proceedings of the Asia-Pacific Conference on Communications*, pp. 620–624, Perth, Australia, October 2005.

[6] L. Von Ahn, M. Blum, and J. Langford, "Telling humans and computers apart automatically," *Communications of the ACM*, vol. 47, no. 2, pp. 56–60, 2004.

[7] Y. Xie and S.-Z. Yu, "A large-scale hidden semi-Markov model for anomaly detection on user browsing behaviors," *IEEE/ACM Transactions on Networking*, vol. 17, no. 1, pp. 54–65, 2009.

[8] Y. Xie, S. Tang, and X. Huang, "Detecting latent attack behavior from aggregated Web traffic," *Computer Communications*, no. 5, pp. 895–907, 2013.

[9] J. Yu, C. Fang, L. Lu et al., "A lightweight mechanism to mitigate application layer DDoS attacks," *Scalable Information Systems*, vol. 18, pp. 175–191, 2009.

[10] P. Du and A. Nakao, "OverCourt: DDoS mitigation through credit-based traffic segregation and path migration," *Computer Communications*, vol. 33, no. 18, pp. 2164–2175, 2010.

[11] H. Beitollahi and G. Deconinck, "Tackling Application-layer DDoS Attacks," *Procedia Computer Science*, vol. 10, pp. 432–441, 2012.

[12] Q.-D. Sun, D.-Y. Zhang, and P. Gao, "Detecting distributed denial of service attacks based on time series analysis," *Chinese Journal of Computers*, vol. 28, no. 5, pp. 767–773, 2005.

[13] R. Yan, Q. Zheng, and H. Li, "Combining adaptive filtering and IF flows to detect DDOS attacks within a router," *KSII Transactions on Internet and Information Systems*, vol. 4, no. 3, pp. 428–451, 2010.

[14] S. Wen, W. Jia, W. Zhou, W. Zhou, and C. Xu, "CALD: Surviving various application-layer DDoS attacks that mimic flash crowd," in *Proceedings of the 4th International Conference on Network and System Security (NSS '10)*, pp. 247–254, Victoria, Australia, September 2010.

[15] S. Haykln, *Adaptive Filter Theory*, Prentice-Hall, Upper saddle River, NJ, USA, 3rd edition, 1996.

[16] J. Viinikka, H. Debar, L. Mé, A. Lehikoinen, and M. Tarvainen, "Processing intrusion detection alert aggregates with time series modeling," *Information Fusion*, vol. 10, no. 4, pp. 312–324, 2009.

[17] J. Platt, "Sequential minimal optimization: a fast algorithm for training support vector machines," Tech. Rep. MSR-TR-98-14, Microsoft Research, 1998.

[18] M. Arlitt and T. Jin, "1998 World Cup Web Site Access Logs," 1998, http://ita.ee.lbl.gov/html/contrib/WorldCup.html.

Hindawi Publishing Corporation
Journal of Control Science and Engineering
Volume 2013, Article ID 462706, 5 pages
http://dx.doi.org/10.1155/2013/462706

Research Article

An Improved Differential Evolution Algorithm Based on Adaptive Parameter

Zhehuang Huang[1,2] and Yidong Chen[2,3]

[1] *School of Mathematical Sciences, Huaqiao University, Quanzhou 362021, China*
[2] *Cognitive Science Department, Xiamen University, Xiamen 361005, China*
[3] *Fujian Key Laboratory of the Brain-Like Intelligent Systems, Xiamen 361005, China*

Correspondence should be addressed to Yidong Chen; ydchen_xm@163.com

Received 2 July 2013; Revised 19 August 2013; Accepted 20 August 2013

Academic Editor: Xiaomei Qi

The differential evolution (DE) algorithm is a heuristic global optimization technique based on population which is easy to understand, simple to implement, reliable, and fast. The evolutionary parameters directly influence the performance of differential evolution algorithm. The adjustment of control parameters is a global behavior and has no general research theory to control the parameters in the evolution process at present. In this paper, we propose an adaptive parameter adjustment method which can dynamically adjust control parameters according to the evolution stage. The experiments on high dimensional function optimization showed that the improved algorithm has more powerful global exploration ability and faster convergence speed.

1. Introduction

In recent years, intelligent optimization algorithms [1] are considered as practical tools for nonlinear optimization problems. Differential evolution algorithm [2, 3] is a novel evolutionary algorithm on the basis of genetic algorithms first introduced by Storn and Price in 1997. The algorithm is a bionic intelligent algorithm by simulation of natural biological evolution mechanism. Its main idea is to generate a temporary individual based on individual differences within populations and then randomly restructure population evolutionary. The algorithm has better global convergence and robustness, very suitable for solving a variety of numerical optimization problems, quickly making the algorithm a hot topic in the current optimization field.

Because it is simple in principle and robust, DE has been applied successfully to all kinds of optimization problems such as constrained global optimization [4], image classification [5], neural network [6], linear array [7], monopoles antenna [8], images segmentation [9], and other areas [10–14].

However, DE algorithm can easily fall into local optimal solution in the course of the treatment of the multipeak and the large search space function optimization problems. In order to improve the optimization performance of the DE, many scholars have proposed many control parameters methods [15, 16]. Although all the methods can improve the standard DE performance to some extent, they still cannot get satisfactory results for some of the functions. In this paper, we propose an adaptive parameter adjustment method according to the evolution stage.

This paper is organized as follows. Related work is described in Section 2. In Section 3 the background of DE is presented. The improved algorithm is presented in Section 4. In Section 5 some experimental tests, results, and conclusions are given. Section 6 concludes the paper.

2. Related Work

The DE algorithm has a few parameters. These parameters have a great impact on the performance of the algorithm, such as the quality of the optimal value and convergence rate. There is still no good way to determine the parameters. In order to deal with this problem, researchers have made some attempts. Gamperle et al. [17] reported that it is more difficult

than expected to choose the control parameters of DE. Liu and Lampinen [18] reported that the performance of DE algorithm is sensitive to the values of the parameters. Different test functions should have different parameter settings.

At present, the parameter settings are mainly three ways:

(1) determined parameter setting method: the method is mainly set by experience, for example, keeping fixed value throughout the entire evolutionary process;

(2) adaptive parameter setting: some heuristic rules are used to modify the parameter values accordingly to the current state;

(3) self-adaptive parameter setting: the idea that "evolution of the evolution" is used to implement the self-adaptive parameter setting.

Liu and Lampinen [19] proposed a fuzzy adaptive parameter setting method which can change the parameters dynamically. The experiment shows the convergence much faster than the traditional DE algorithm when adapting F and CR. In [20], self-adapting control parameters in DE are proposed; the results show the improved algorithm is better than, or at least comparable to, the standard DE algorithm.

3. Introduction to DE

Compared to other evolutionary algorithms, DE reserves population-based global search strategy and uses a simple mutation operation of the differential and one-on-one competition, so it can reduce the genetic complexity of the operation. At the same time, the specific memory capacity of DE enables it to dynamically track the current search to adjust their search strategy with a strong global convergence and robustness. So it is suitable for solving some of the complex environments of the optimization problem. Basic operations such as selection, crossover, and mutation are the basis of the difference algorithm.

In an iterative process, the population of each generation G contains N individuals. Suppose that the individual i of generation G is represented as

$$X_{iG} = \left(x_{iG}^1, x_{iG}^2, \ldots, x_{iG}^D\right), \quad i = 1, 2, \ldots. \tag{1}$$

3.1. Mutation Operation. An individual can be generated by the following formula:

$$X_{r_1,G+1} = X_{r_1,G+1} + F * \left(X_{r_2,G} - X_{r_3,G}\right). \tag{2}$$

Here r_1, r_2, and r_3 are random numbers generated within the interval $[1, N]$ and variation factor F is a real number of the interval $[0, 2]$; it controls the amplification degree of the differential variable $X_{r_2,G} - X_{r_3,G}$.

3.2. Crossover Operation. In difference algorithm, the cross-operation is introduced to the diversity of the new population. According to the crossover strategy, the old and new individual exchange part of the code to form a new individual. New individuals can be represented as follow:

$$X_{i,G+1} = \left(x_{1i,G+1}, x_{2i,G+1}, \ldots, x_{Di,G+1}\right), \quad i = 1, 2, \ldots, \tag{3}$$

where

$$x_{ji,G+1} = \begin{cases} V_{ji,G+1}, & \text{if } \left(\text{rand}b\left(j\right) \le \text{CR}\right) \text{ or } \left(j = mbr\left(i\right)\right), \\ V_{ji,G+1}, & \text{if } \left(\text{rand}b\left(j\right) > \text{CR}\right) \text{ and } \left(j \ne mbr\left(i\right)\right) \\ & \left(j = 1, 2, \ldots, D\right), \end{cases} \tag{4}$$

where rand $b(j)$ is uniformly distributed in the interval $[0, 1]$ and CR is crossover probability in the interval $[0, 1]$. $mbr\,(i)$ means a random integer between $[0, D]$.

3.3. Selection Operation. Selection operation is greedy strategy; the candidate individual is generated from mutation and crossover operation competition with target individual:

$$x_{i,G+1} = \begin{cases} U_{i,G}, & \text{if } \left(f\left(U_{i,G}\right) > f\left(x_{i,G+1}\right)\right), \\ x_{i,G+1}, & \text{if } \left(f\left(U_{i,G}\right) \le f\left(x_{i,G+1}\right)\right), \end{cases} \tag{5}$$

where f is the fitness function.

The basic differential evolution (DE) algorithm is shown as Algorithm 1.

Algorithm 1 (the differential evolution algorithm). (1) Initialize the number of population NP, the maximum number of evolution Maxinter, the scale factor and cross-factor.

(2) Initialize the population pop.

(3) Follow the DE/rand/1/bin policy enforcement options, and produce a new generation of individual:

(a) mutation operation;

(b) crossover operation;

(c) selection operation.

(4) Until the termination criterion is met.

The flow chart of differential evolution algorithm is shown in Figure 1.

4. The Adaptive Control Parameter Adjustment Method (ADE)

From standard DE algorithm, it is known that scale factor F and cross-factor CR will not only affect convergence speed of the algorithm, but may also lead to the occurrence of premature phenomenon. In this paper, we propose an adaptive adjustment method according to the evolution stage.

We use a sine function (1/4 cycle) with value of $(-1, 0)$ and a cosine function (1/4 cycle) with value of $(0, 1)$. The image of the two functions shows slower change at the beginning and in the end, with rapid changes and gradual increase in the middle. It is very suitable for setting F value and CR value.

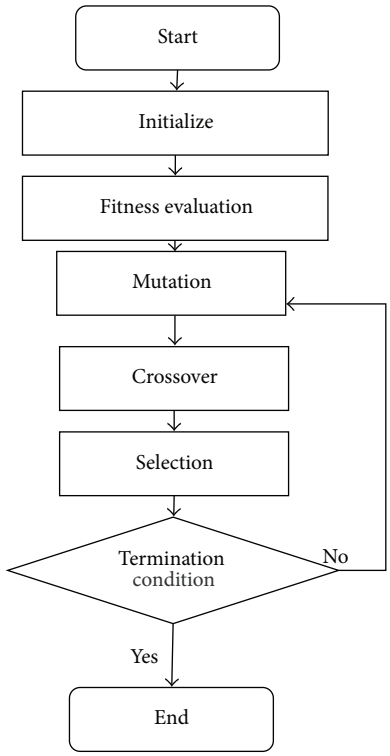


Figure 1: Flow chart of difference evaluation algorithm.

Table 1: Functions used to test the effects of ADE.

Function	Function expression
Sphere function	$f_1(x) = \sum_{t=1}^{n} x_t^2$
Rastrigin function	$f_2(x) = \sum_{t=1}^{n} (x_t^2 - 10\cos(2\pi x_t) + 10)$
Griewank function	$f_3(x) = \frac{1}{4000}\sum_{t=1}^{n}(x_t - 100)^2 - \prod_{t=1}^{n}\cos\left(\frac{x_t - 100}{\sqrt{t}}\right) + 1$
Ackley function	$f_4(x) = 20 + e - 20e^{-0.2\sqrt{(\sum_{t=1}^{n} x_t{}^2)/n}} - e^{(\sum_{t=1}^{n}\cos(2\pi x_t))/n}$
Shaffer's function	$f_5(x) = 0.5 + \frac{\left(\sin\sqrt{x_1^2 + x_2^2}\right)^2 - 0.5}{\left(1 + 0.001\left(x_1^2 + x_2^2\right)\right)^2}$

Table 2: The performances of DE and ADE.

Function	DE		ADE	
	Optimal	Time (s)	Optimal	Time (s)
Sphere	0.039	0.42	0.019	0.31
Rastrigrin	19.07	0.43	4.75	0.36
Griewank	0.43	0.45	0.33	0.28
Ackey	1.16	0.45	1.31	0.27
Shaffer	0.00973	0.43	0.00973	0.35

The early stage and the late stage of scale factor F and cross-factor CR are relatively small, with relatively fast increase in the middle, just to meet the global search of PE

$$F = \begin{cases} \alpha + (1-\alpha) \times \sin\left(\dfrac{\pi t}{\text{MAXITER}} - \dfrac{\pi}{2}\right), \\ \qquad \text{if}\left(t \le \dfrac{\text{MAXITER}}{2}\right), \\ \alpha - (1-\alpha) \times \cos\left(\dfrac{\pi}{2} - \dfrac{\pi t}{\text{MAXITER}}\right), \\ \qquad \text{otherwise}, \end{cases} \tag{6}$$

$$\text{CR} = \begin{cases} \beta + (1-\beta) \times \sin\left(\dfrac{\pi t}{\text{MAXITER}} - \dfrac{\pi}{2}\right), \\ \qquad \text{if}\left(t \le \dfrac{\text{MAXITER}}{2}\right), \\ \beta - (1-\beta) \times \cos\left(\dfrac{\pi}{2} - \dfrac{\pi t}{\text{MAXITER}}\right), \\ \qquad \text{otherwise}, \end{cases} \tag{7}$$

where α and β are constants; for example, we can set $\alpha = 0.8$, and $\beta = 0.75$ in the experiment. MAXITER is the maximum number of iterations, and t is the current number of iterations.

The procedure for implementing the APE is given by the following steps.

Algorithm 2 (the improved differential evolution algorithm). (1) Initialize the number of population NP, the maximum number of evolution Maxinter, scale factor F and cross-factor CR.

(2) Initialize the population pop.

(3) Update the scaling factor F of each individual according to the above formula (6).

(4) Update the cross-factor CR of each individual according to the above formula (7).

(5) Perform the following behavior: Mutation, Crossover and Selection, and produce a new generation of individuals.

(6) Until the termination criterion is met.

5. Experimental Results

A set of unconstrained real-valued benchmark functions shown in Table 1 was used to investigate the effect of the improved algorithm.

The results are shown in Table 2. Each point is made from average values of over 10 repetitions. We set scale factor $F = 0.6$ and cross-factor CR = 0.5 for the standard PE algorithm and dynamically adjust F and CR according to the evolution stage for the ADE algorithm.

From Table 2, we can see that no algorithm performs better than the others for all five functions, but on average, the ADE is better than DE algorithm.

For Sphere function, Rastrigin function, and Griewank function, ADE algorithm can effectively improve the accuracy such that the optimal value obtained is much closer to the theoretical one compared with the standard DE algorithm. Ackley function is a multimodal function; from the results of iteration, the accuracy of the improved algorithm is not as that of good as the standard DE algorithm,

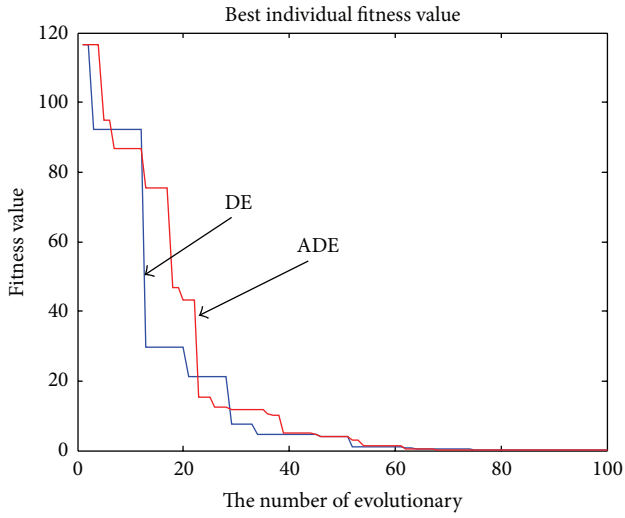


Figure 2: Sphere function.

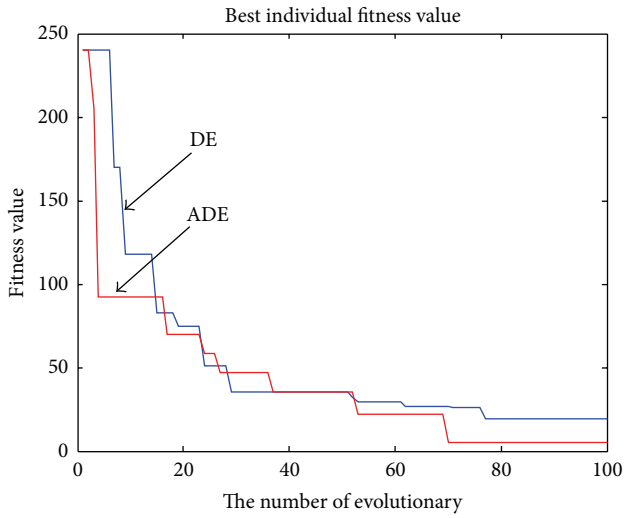


Figure 3: Rastrigin function.

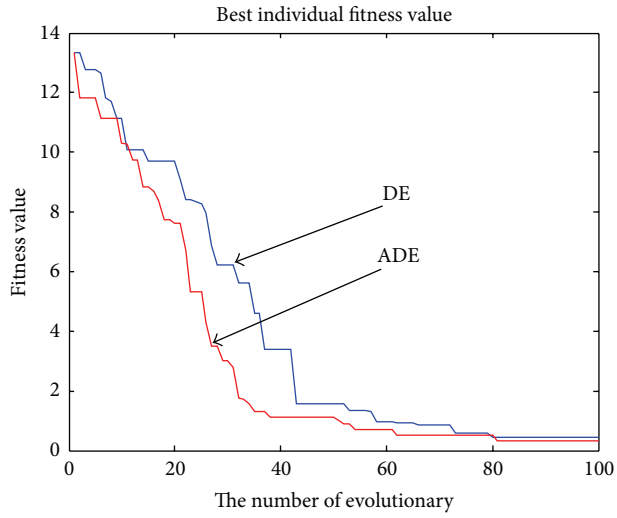


Figure 4: Griewank function.

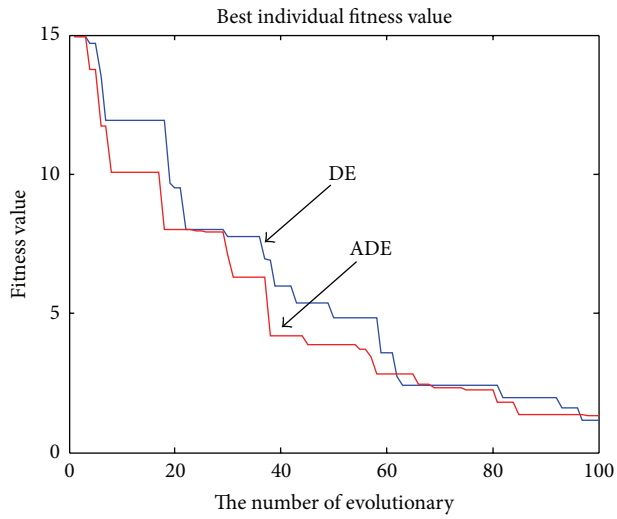


Figure 5: Ackley function.

but the difference is small and acceptable. For Shaffer function, there is no obvious superior algorithm.

For all the five functions, there is a significant improvement as expected on the convergence time. These experimental results show that improving the algorithm can effectively improve the convergence speed with excellent convergence effect.

The comparison of two methods with convergent curves is shown in Figures 2, 3, 4, 5, and 6. The experiment results show the ADE algorithm has better result. Compared with DE, the ADE algorithm has both global search ability and fast convergence speed.

6. Conclusion

The scale factor F and cross-factor CR have a great impact on the performance of the algorithm, such as the quality of the optimal value and convergence rate. There is still no

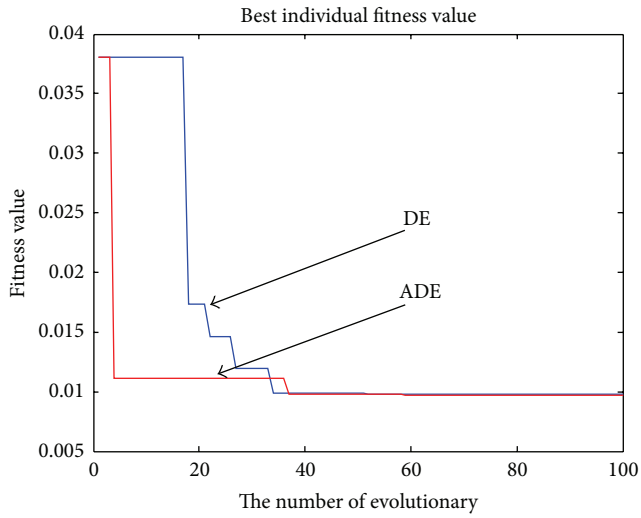


Figure 6: Shaffer's function.

good way to determine the parameters. In this paper, we propose an adaptive parameter adjustment method according to the evolution stage. From before mentioned experiment, we can know the improved algorithm has more powerful global exploration ability and faster convergence speed and can be widely used in other optimization tasks.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant no. 61005052), the Fundamental Research Funds for the Central Universities (Grant no. 2010121068), and the Science and Technology Project of Quanzhou (Grant no. 2012Z91).

References

[1] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multi-dimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, pp. 58–73, 2002.

[2] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[3] R. Storn and K. Price, "Differential evolution for multi-objective optimization," *Evolutionary Computation*, vol. 4, pp. 8–12, 2003.

[4] H. K. Kim, J. K. Chong, K. Y. Park, and D. A. Lowther, "Differential evolution strategy for constrained global optimization and application to practical engineering problems," *IEEE Transactions on Magnetics*, vol. 43, no. 4, pp. 1565–1568, 2007.

[5] M. G. H. Omran and A. P. Engelbrecht, "Self-adaptive differential evolution methods for unsupervised image classification," in *Proceedings of the IEEE Conference on Cybernetics and Intelligent Systems*, pp. 1–6, Bangkok, Thailand, June 2006.

[6] H. Dhahri and A. M. Alimi, "The modified differential evolution and the RBF (MDE-RBF) neural network for time series prediction," in *Proceedings of the International Joint Conference on Neural Networks 2006 (IJCNN '06)*, pp. 2938–2943, Vancouver, Canada, July 2006.

[7] S. Yang, Y. B. Gan, and A. Qing, "Sideband suppression in time-modulated linear arrays by the differential evolution algorithm," *IEEE Transactions on Antennas and Propagations Letters*, vol. 1, no. 1, pp. 173–175, 2002.

[8] A. Massa, M. Pastorino, and A. Randazzo, "Optimization of the directivity of a monopulse antenna with a subarray weighting by a hybrid differential evolution method," *IEEE Transactions on Antennas and Propagations Letters*, vol. 5, no. 1, pp. 155–158, 2006.

[9] V. Aslantas and M. Tunckanat, "Differential evolution algorithm for segmentation of wound images," in *Proceedings of the IEEE International Symposium on Intelligent Signal Processing (WISP '07)*, Alcalá de Henares, Spain, October 2007.

[10] L. H. Wu, Y. N. Wang, X. F. Yuan, and S. W. Zhou, "Differential evolution algorithm with adaptive second mutation," *Chinese Journal of Control and Decision*, vol. 21, no. 8, pp. 898–902, 2006.

[11] C. T. Su and C. S. Lee, "Network reconfiguration of distribution systems using improved mixed-integer hybrid differential evolution," *IEEE Transactions on Power Delivery*, vol. 18, no. 3, pp. 1022–1027, 2003.

[12] M. F. Tasgetiren, P. N. Suganthan, T. J. Chua, and A. Al-Hajri, "Differential evolution algorithms for the generalized assignment problem," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 2606–2613, Trondheim, Norway, May 2009.

[13] W. G. Zhang, H. P. Chen, D. Lu, and H. Shao, "A novel differential evolution algorithm for a single batch-processing machine with non-identical job sizes," in *Proceedings of the 4th International Conference on Natural Computation (ICNC '08)*, pp. 447–451, Jinan, China, October 2008.

[14] T. Sum-Im, G. A. Taylor, M. R. Irvings, and Y. H. Song, "A differential evolution algorithm for multistage transmission expansion planning," in *Proceedings of the 42nd International Universities Power Engineering Conference (UPEC '07)*, pp. 357–364, Brighton, UK, September 2007.

[15] Z. F. Wu, H. K. Huang, B. Yang, and Y. Zhang, "A modified differential evolution algorithm with self-adaptive control parameters," in *Proceedings of the 3rd International Conference on Intelligent System and Knowledge Engineering (ISKE '08)*, pp. 524–527, Xiamen, China, November 2008.

[16] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing*, vol. 9, no. 6, pp. 448–462, 2005.

[17] R. Gamperle, S. D. Muller, and P. Koumoutsakos, "A parameter study for differential evolution," in *Proceedings of the International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation (WSEAS '02)*, pp. 11–15, Interlaken, Switzerland, February 2002.

[18] J. Liu and J. Lampinen, "On setting the control parameter of the differential evolution method," in *Proceedings of the 8th Internatuional Conference on Soft Computing (MENDEL '02)*, pp. 11–18, Brno, Czech Republic, June 2002.

[19] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Computing*, vol. 9, no. 6, pp. 448–462, 2005.

[20] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.