# A Bio-Inspired Approach to Task Assignment of Swarm Robots in 3-D Dynamic Environments

Xin Yi, Anmin Zhu, *Member, IEEE*, Simon X. Yang, *Senior Member, IEEE*, and Chaomin Luo, *Member, IEEE*

***Abstract*—Intending to mimic the operating mechanism of biological neural systems, a self organizing map-based approach to task assignment of a swarm of robots in 3-D dynamic environments is proposed in this paper. This approach integrates the advantages and characteristics of biological neural systems. It is capable of dynamically planning the paths of a swarm of robots in 3-D environments under uncertain situations, such as when some robots are presented in or broken down or when more than one robot is needed for some special task locations. A Bezier path optimizing algorithm and a parameter adjusting algorithm are integrated in this paper. It is capable of reducing the complexity of the robot navigation control and limiting the number of convergence iterations. The simulation results with different environments demonstrate the effectiveness of the proposed approach.**

***Index Terms*—3-D environments, bio-inspired approach, swarm robots, task assignment.**

## I. Introduction

Multirobot system, such as a swarm of robots, a group of unmanned air vehicles, or a group of missiles, is a relatively new research area, which is derived from the research and development of a single robotic system [1], [2]. The research direction of multirobot system focuses more on robots with individual simplicity, large-scale in number, cooperative ability, and environmental adaptability. Its applications are extended in a variety of fields [3]. The U.S. takes the lead in conducting research in military applications. U.S. ARMY Research Laboratory invests $38 million on the research of miniature robot group in order to raise the army's fighting capacity [4]. MIT has designed a task-level control system for multiple unmanned air vehicles [5]. Additionally, multirobot system for tracking and capturing multitarget has also been used in disaster relief, and environmental monitoring [6], [7]. Multirobot system has found an increasingly wide utilization in all fields, and has become one of the hot spots of robotics research domain.

Manuscript received December 16, 2013; revised June 21, 2015 and January 22, 2016; accepted February 12, 2016. Date of publication March 15, 2016; date of current version March 15, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61273354, and in part by the Natural Sciences and Engineering Research Council of Canada. This paper was recommended by Associate Editor F. Karray. *(Corresponding author: Anmin Zhu.)*

X. Yi and A. Zhu are with the School of Computer and Software, Shenzhen University, Shenzhen 518060, China (e-mail: azhu@szu.edu.cn).

S. X. Yang is with the ARIS Laboratory, School of Engineering, University of Guelph, Guelph, ON N1G 2W1, Canada.

C. Luo is with the Robotics and Intelligent Systems Laboratory, Department of Electrical and Computer Engineering, University of Detroit Mercy, Detroit, MI 48221-3038 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2016.2535153

Due to the complexity of the multirobot system, and the fact that researchers come from different disciplines, it is reasonable that the research has different directions, including architectural design of multirobot system, sensor network research of mobile robots, communication protocol design of multirobots, cooperative control design of multirobots, and research of swarm robot intelligence [8]–[12]. Among these, swarm robot collaborative-control is one of the key research directions [13]. Collaboratively controlling a swarm of robots to capture multiple moving targets in variable unknown environments is an instance of this research direction. Specifically, the aim is to control a swarm of robots with cooperation among robots. The robots are able to collaborate expeditiously to move to the targets' positions without collision or severe competition [14], [15]. However, because of the large amount of robots, the uncertainty of moving target, and the unknown of external environments, this paper on swarm robot control encounters huge challenges. The challenges include a complicated and robust control algorithm to adapt task allocation with its own path planning, collision avoidance, and motion control. This cannot be completed by simple summation of individual robot functions. To control a swarm of robots, it is not sufficient to adapt the robotic motion only according to the current status of environments. The situations of the swarm of robots, and the task targets also have to be considered to conduct behaviors of cooperation and coordination. Actually, task assignment of a swarm robot system is an NP problem [16]–[18].

There have been some studies on the task assignment of multirobot systems. It can be broadly divided into two categories: 1) classic traditional algorithm and 2) intelligent algorithm. Traditional methods create control model based on the current information of static environments, and then find optimal paths using conventional search methods, such as gradient algorithm, A* algorithm, random search, and enumeration methods [13]. These traditional methods have obvious disadvantages, such as the presence of local minimums, high computing cost, and unsolved high-dimensional problems. Furthermore, with the increase in number of robots, the computational complexity increases rapidly. To solve the rapidly increasing complexity, improved traditional methods [19], [20] and intelligent algorithms including evolutionary computing methods [21], swarm intelligence [22], neural computing methods [15], [23], and biological neural network solutions [24]–[26] are proposed. Brass *et al.* [19] proposed a concurrent path search technology for multirobot systems in a tree and graphic environment. It can greatly improve computational efficiency because the path search method for multirobot system is simultaneous. Kotb *et al.* [20] proposed a workflow-based framework for the cooperation of multi-robots. This algorithm can obtain the minimum cost of the path planning. However, these two multirobot path planning methods are modified from the traditional method by simply improving the path search strategy. These two methods can apply only to specific environments, but not to complicated and dynamic environments. Zhong *et al.* [21] proposed a task assignment algorithm, which can make multiple aerial vehicles attack targets with dynamic values. This method combined multisubgroup ant colony optimization algorithm to a planning algorithm for multidestination routes, which can obtain a reasonable result in a short time. However, this method cannot apply to dynamic environments. Jevtic *et al.* [22] proposed a distributed bee algorithm. This algorithm can run rational and efficient task allocation regarding the efficiency and quality of planned paths. However, it does not have avoidance mechanisms and cannot achieve the requirements of real-time, online planning and allocation. Zhu and Yang [14] proposed a self organizing map (SOM)-based neural network, that can well cope with the path planning problem of multirobot systems. It can make real-time path planning in complex and dynamic environments. Their extended work [15] considered the factor of the robot's orientation. The computational complexity of the algorithm is low, and there is no "curse of dimensionality" problem. However, the parameters of this algorithm are dependent and sensitive, which can easily result in misconvergence problem during the path planning iteration process. Zhu *et al.* [23] improved the SOM-based neural network method for dynamic task assignment of underwater robots on 3-D environment, which has low computational complexity, high fault tolerance, and adaptability. However, this method lacks fairness and load balance so that regnant robot will take excessive restricted resources. Ni and Yang [24] proposed a biological neural network-based model to solve the multirobot hunting problem. Li *et al.* [25] also proposed a biological neural network-based coordinated hybrid agent framework for multirobot path planning. These two methods have lower computational cost and are capable of real-time and on-line path planning in complicated and dynamic environments. However, these two methods have computing redundancy and storage redundancy problems, so these cannot be applied to high dimensional and large scale environments.

Most of the multirobot systems are considered in 2-D environments. However, current applications, such as scientific deep underwater robotic systems [23], [27], unmanned aerial vehicles [28], [29], and missile defense systems [30], must consider high dimensional space, especially in 3-D environments. Multirobot systems urge on-line path planning methods in complicated, robust, fault tolerant, highly dynamic, and highly dimensional environments.

In this paper, inspired by the self-organizing, self-adaptive, self-repairing, self-perfecting, and self-optimizing characteristics of biological systems, we intend to imitate biological mechanisms and neural system functions, apply the advantages and characteristics of the biological neural system to swarm robot system using SOM-based neural network approach, and then identify the path planning of a swarm of robots with cooperative processing capability among robots in dynamic 3-D environments. Based on previous research [14], [15], [24], three aspects are different or improved in this paper.

1) We complete the modeling of multirobot system in 3-D space and proposed the task assignment algorithm for swarm robots in 3-D environments in this paper, but such system only considered 2-D environments in our previous works.
2) New optimization methods are proposed to adjust the learning rate, which is able to reduce the parameter dependency and solve the misconvergence problem occurred in our previous works.
3) Bezier-based optimization methods are applied to smooth and stabilize the planned paths in this paper, but our previous work cannot obtain a set of smooth and steady paths, which increase the complexity of robotic control.

The rest of this paper is organized as follows. In Section II, the proposed bio-inspired approach to task assignment of swarm robots in 3-D environments is presented. The simulation results and analysis on task assignment of swarm robots in 3-D environments are presented in Section III. A further discussion on simulation results about the path optimization and parameter adjustment are given in Section IV. Finally, concluding remarks and future work are presented in Section V.

## II. Proposed Approach

In this section, the proposed approach is introduced elaborately. It is divided into five subsections.

1) *Introduction of SOM:* This section depicts the basic theory of SOM algorithm.
2) *Environmental Representation of Robots:* This section addresses the representation of swarm robots in 2-D and 3-D environments.
3) *Algorithm of Path Planning:* This section described the SOM-based planning algorithm.
4) *Adjustment and Optimization of Parameters:* This section introduces the drawbacks of the conventional method in parameter-adjustment and the superiority of the improved method in parameter-adjustment.
5) *Optimization of Bezier Curving:* This section presents the correction and adjustment of path in the optimization of Bezier curving (OBC).

### A. Introduction of SOM

Inspired by the ubiquity of cortical maps in the central neural system, SOM algorithm was first introduced by Kohonen in the 1980s [31], and extended later [32], [33]. SOM algorithm is based on the idea that there is a meaningful order of processing units in the mammalian brain, where each part

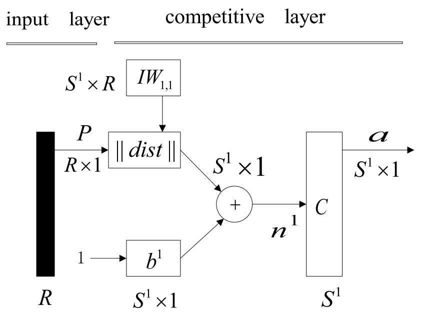


Fig. 1. Model of SOM.

is dedicated to a specific task and each group of neurons is sensitive to a particular type of input signal. The units are determined by parameters which can be changed in certain processes to produce meaningful organizations [14]. Because of its universal applicability, SOM algorithm became a valuable tool and was applied to unsupervised learning, clustering, and other issues. The mathematical model of SOM is shown in Fig. 1, where ||dist|| is the calculating part, $P$ is the input vector, IW is the weight matrix, $b$ is the threshold vector, $R$ and $S^1$ are dimensions of the vectors, and $C$ is the competitive layer.

According to the description in Fig. 1, the competitive layer will obtain a winner neuron. After that, both weights and thresholds of the SOM structure are adjusted by Kohonen rule in the neural network. The rule of weight adjustment is described as

$$IW_i(t) = IW_i(t-1) + a(t-1)[P(t-1) - IW_i(t-1)], i \in 1, 2 \ldots S^1 \quad (1)$$

where $IW_i(t)$ is the weight of the $i$th neuron in the $t$th calculating process, $a(t-1)$ is the result vector in the $t$th calculating process, $P(t-1)$ is the input vector in the $t$th calculating process, and $i$ is the serial number of neuron.

## B. Environmental Representation of Robot

The omni-directional mobile robot in 2-D space can be described mathematically as a vector $(x, y, \theta)$, where $x$ is the coordinate value in the $x$-axis, $y$ is the coordinate value in the $y$-axis, and $\theta$ is the angle between the $x$-axis and the direction of the robot. The mathematical representation of omni-directional mobile robots in 2-D space are described as follows:

$$\overrightarrow{r_i} = (x_i, y_i, \theta_i), \theta_i \in (-\pi, \pi); i = 1 \ldots n \quad (2)$$

where $i$ is the index of the robot and $n$ is the number of robots.

The omni-directional mobile robot in 3-D space is different, which can be described mathematically as a vector $(x, y, z, \alpha, \beta)$, where $x$ is the coordinate value in the $x$-axis, $y$ is the coordinate value in the $y$-axis, $z$ is the coordinate value in the $z$-axis, $\alpha$ is the angle between the direction of the robot and the $XOY$ plane, and $\beta$ is the angle between projection of direction of robot in the $XOY$ plane and the $x$-axis.

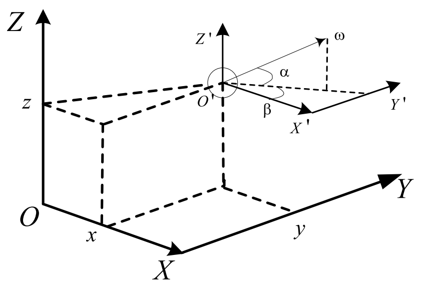


Fig. 2. Attributes of robot.

The mathematical attributes of omni-directional mobile robot in 3-D space are described as follows:

$$\overrightarrow{r_i} = (x_i, y_i, z_i, \alpha_i, \beta_i), \alpha_i \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right), \beta_i \in (-\pi, \pi); \quad i = 1 \ldots n \quad (3)$$

where $i$ is the index of the robot and $n$ is the number of robots. The attributes of a robot are shown in Fig. 2.

The targets have some attributes in 3-D space are described as follows:

$$\overrightarrow{t_j} = (x_j, y_j, z_j) \quad j = 1 \ldots m \quad (4)$$

where $j$ is the index of a target, and $m$ is the number of targets.

## C. Algorithm of Path Planning

The proposed path planning algorithm is based on SOM neural network, which has more efficiency, fault tolerance, and robustness. It can plan paths for a swarm of robots online. Details of the path planning algorithm are explained as follows.

*1) Model of the Input Layer:* There are three input neurons in the input layer, which represent $x_j$, $y_j$, and $z_j$ of the vector described in (4), respectively. The winning rules are described as

$$\overrightarrow{T_k} = \left[\overrightarrow{t_j}\right], j \in [1, m] \Leftarrow \text{random}\left\{\overrightarrow{t_j} = (x_j, y_j, z_j); j = 1 \ldots m\right\}, k = 1 \ldots \quad (5)$$

where $\overrightarrow{T_k}$ is the $k$th input vector, which is selected randomly from all of the target vectors and includes the same attributes of the selected target vector.

*2) Model of the Competitive Layer:* Each neuron represents a robot in the competitive layer, respectively, which includes the same attributes of the relative robot. The winning rules are described as

$$\overrightarrow{R_k} = \left[\overrightarrow{r_i}\right], i \in [1, n] \Leftarrow \min\left\{\text{Dist}\left(\overrightarrow{T_k}, \overrightarrow{r_i}, \omega\right), i = 1 \ldots n\right\}, k = 1 \ldots \quad (6)$$

where $\overrightarrow{R_k}$ is the $k$th result vector, $\overrightarrow{T_k}$ is the $k$th input vector, $\omega$ is the weight parameter, and $\text{Dist}(\overrightarrow{T_k}, \overrightarrow{r_i}, \omega)$ is a distance function described as follows:

$$\text{Dist}\left(\overrightarrow{T_k}, \overrightarrow{r_i}, \omega\right) = |\overrightarrow{T_k} - \overrightarrow{r_i}| \quad (7)$$

$$|\overrightarrow{T_k} - \overrightarrow{r_i}| = \sqrt{(x_j - x_i)^2} + \sqrt{(y_j - y_i)^2} + \sqrt{(z_j - z_i)^2} + \omega \times \Delta \Phi_{ij}, i \in 1, 2 \ldots n, j \in 1, 2 \ldots m \quad (8)$$

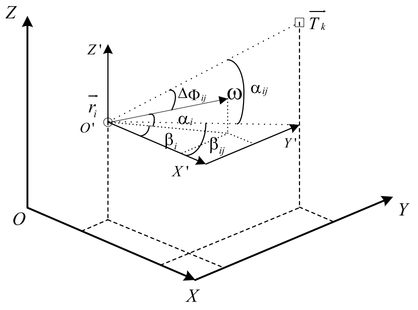


Fig. 3. Angles of robot in 3-D space.

where $j$ is the index of input vector, $i$ is the index of the robot, and $\Delta\Phi_{ij}$ is the angle between direction of the robot and direction of the target.

Fig. 3 demonstrates the related angles. The details of these angles are defined as

$$\Delta\Phi_{ij} = \arccos\left(\cos\left(\alpha_{ij}-\alpha_i\right)\cos\left(\beta_{ij}-\beta_i\right)\right)$$
$$i = 1\ldots n, \alpha_{ij}\in\left(-\frac{\pi}{2},\frac{\pi}{2}\right), \beta_{ij}\in(-\pi,\pi) \quad (9)$$

$$\alpha_{ij} = \mathrm{artan}\left(\frac{z_j-z_i}{\sqrt{\left(x_j-x_i\right)^2+\left(y_j-y_i\right)^2}}\right), \alpha_{ij}\in\left(-\frac{\pi}{2},\frac{\pi}{2}\right) \quad (10)$$

$$\beta_{ij} = \mathrm{ar}\sin\left(\frac{y_j-y_i}{\sqrt{\left(x_j-x_i\right)^2+\left(y_j-y_i\right)^2}}\right), \beta_{ij}\in(-\pi,\pi) \quad (11)$$

where $\alpha_{ij}$, $\alpha_i$, $\beta_{ij}$, and $\beta_i$ are shown in Fig. 3, $j$ is the index of the input vector, and $i$ is the index of the robot.

*3) Parameter Adjustment Rules:* Parameter adjustment rules can be divided into weight adjustment, learning rate adjustment, and scope adjustment.

The weight adjustment method is described in (12). When the distance between the winning neuron and the input vector is close sufficiently, the winning neuron is adjusted to the input vector. Otherwise, the second equation will be adopted

$$\overrightarrow{r_{i,k+1}} = \begin{cases} \overrightarrow{T_k} & \mathrm{Dist}\left(\overrightarrow{T_k},\overrightarrow{r_{i,k}},\omega\right) < \varsigma\min \\ \overrightarrow{r_{i,k}} + \lambda(k,\eta)f\left(\overrightarrow{r_{i,k}},\overrightarrow{R_k},k,G\right)\left(\overrightarrow{T_k}-\overrightarrow{r_{i,k}}\right) & \text{other} \end{cases} \quad (12)$$

where $\varsigma$ is convergence parameters, min is the minimal distance among robots, $\lambda(\eta,k)$ is the function of the learning rate, and $f(\overrightarrow{r_{i,k}},\overrightarrow{R_k},k,G)$ is the function of the scope.

The learning rate adjustment method is described as

$$\lambda(k,\eta) = (1-\mathrm{dec})^{k-1}\eta \quad (13)$$

where $k$ is the iterations, $\eta$ is the initial learning rate, and dec is the rate of descent. Details of the improved adjustment of learning rate is discussed in Section II-D.

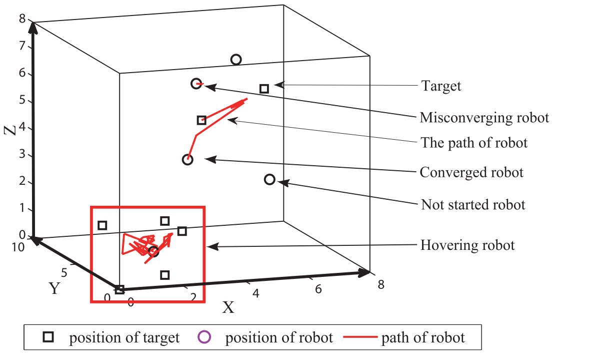


Fig. 4. Misconvergence problem by using traditional adjustment of learning rate.

The scope adjustment method is described as

$$f\left(\overrightarrow{r_{i,k}},\overrightarrow{R_k},k,G\right) = e^{-\frac{\left(\overrightarrow{R_k}-\overrightarrow{r_{i,k}}\right)^2}{(1-\mathrm{dec})^{k-1}G}} \quad (14)$$

where $\overrightarrow{r_{i,k}}$ is the vector of the $i$th robot in the $k$th iteration, $\overrightarrow{R_k}$ is the vector of the winning robot in the $k$th iteration, dec is the rate of descent, and $G$ is the initial scope. Equation (14) states that the affected scope of the winning neurons decreases gradually along with the increase of iterations.

## D. Optimization of Adjustment Parameters

In traditional parameter adjustment methods, the learning rate is decreased in accordance with a certain rate, which causes bad results for the swarm robot systems. Assume that a robot is surrounded by several targets and some other robots are far away from these targets, then the closer robot is hovering among these targets while others cannot participate in the competition. These are severe drawbacks of traditional parameter adjustment methods for path planning as shown in Fig. 4. Some sequent influences are issued.

1) The robots are hovering among the multiple targets, This makes the paths complicated and increases the complexity of robotic control.
2) It is easy for the "Hovering" robot to be the winner, while others are difficult because they are far away from the target. It is unfair for the competition and unbalanced for the whole system.
3) Because the Hovering robots win frequently, the number of iterations is increased without control. Therefore, the convergence of the task assignment for swarm robots becomes uncertain.

Because there are more parameters and more complex topologies in high dimensional space than in low-dimensional space, these problems occur far more frequently in the high-dimensional space than in low-dimensional space. It means that the probability of these problem occurring in 3-D space is much higher than that in 2-D space. In addition, because the misconvergence of the path planning occurs easily and the parameters are required stringently, the traditional learning rate adjustment method (13) is sensitive to the descent rate and the number of iteration.

In traditional methods for adjusting the learning rate (13), the initial assumption is $\eta$, and $N$ is the summation of robot iterations, described in the following equation:

$$\begin{aligned}\eta + (1-\eta)\eta + (1-(1-\eta)\eta-\eta)\eta \ldots + (1-\eta)^{N-1}\eta \\ = \frac{\eta\left(1-(1-\eta)^N\right)}{1-(1-\eta)} \\ = \left(1-(1-\eta)^N\right).\end{aligned} \tag{15}$$

Therefore, only if (16) is true, the robot can reach at the target

$$\left(1-(1-\eta)^N\right) < 1-\varsigma \tag{16}$$

where $\varsigma$ is the convergence parameter in (12). Then, $N > \log_{(1-\eta)}\varsigma$ can be deduced from (16).

To solve these problems, two improved methods for adjusting the learning rate are proposed [31], defined as the following equations:

$$\lambda(k,\eta) = \begin{cases}\eta\left(\frac{k}{nN\delta}-1\right)^2 & 0 < k < nN\delta \\ \eta & k > nN\delta\end{cases} \tag{17}$$

and

$$\lambda(k,\eta) = \begin{cases}\eta\left[1-\sin\left(\frac{\pi k}{nN\delta}\right)\right] & 0 < k < nN\delta \\ \eta & k > nN\delta\end{cases} \tag{18}$$

where $k$ is the number of iterations, $n$ is the number of robots, and $\delta$ is a parameter. Its value is represented by the following equations, respectively. $\delta = (2/\int_0^2 (x-1)^2 dx) = 3$, and $\delta = (\pi/\int_0^\pi (1-\sin x)dx) = (\pi/\pi-2)$. One of the two improved methods can be used to replace (13). Compared with the traditional method which adjusts the learning rate (13), the improved methods in (17) and (18) do not need to set the reducing parameter dec, and can guarantee that the path planning can converge under the constant number of iterations.

## E. Optimization of Bezier Curving

Because the above algorithm produces a set of unsmoothed and concussive paths when planning the paths of a swarm of robots, the complexity of the control of robots will be substantially increased. Therefore, a Bezier curving optimization method is integrated in the algorithm to smooth the paths.

It is commonly known that the original path before optimization is composed of several segments in the 3-D environment, which is described by piecewise function

$$\text{Path}(l) = \begin{cases}X(l) \\ Y(l) & 0 \le l \le L \\ Z(l)\end{cases} \tag{19}$$

where $l$ is the path length of path range from origin to current point and $L$ is the total length of the planning path.

The Bezier curving is described as

$$B(t) = \sum_{i=0}^{n} P_i b_{i,n}(t),\ t \in [0,1] \tag{20}$$

where $b_{i,n}(t)$ is the $n$th order Bernstein basement polynomial, which is defined as

$$b_{i,n}(t) = \binom{n}{i} t^i (1-t)^{n-i},\ i = 0,\ldots n \tag{21}$$

and $P_i$ is the $i$th control point, which is defined as

$$P_i = (x_i, y_i, z_i). \tag{22}$$

To keep the coherence between the un-optimized path and the optimized one, all control points must be selected from the un-optimized path during the process of Bezier curving optimization, i.e., $P_i = \text{Path}(l_i)$. The choice of the control point $P_i$ will directly affect the optimization performance. Therefore, three different methods of choosing the control point are introduced as follows.

*1) Segmental Points:* The first method of choosing the control points is named segmental points (SGPs). Assume that there are some un-optimized paths $l_i$ $(i \in 1,2,\ldots,n \wedge 0 \le l_i \le L)$, and it satisfies the equation $(d\text{Path}(l)/dl)|_{l=l_i^+} \ne (d\text{Path}(l)/dl)|_{l=l_i^-}$. Then we set $A = \{\text{Path}(l_i)\ i = 1,2,\ldots,n \wedge 0 \le l_i \le L\}$, which includes all the SGP on the function Path($l$), where $n$ is the number of set. The definition of segmental point is described as

$$P_i = \text{Path}(l_i) = (X(l_i), Y(l_i), Z(l_i)) \wedge P_i \in A, i = 1,2,\ldots n \tag{23}$$

which guarantees that the control points $P_i$ are selected from the un-optimized path Path($l$).

*2) Equal-Length Point:* The second method of choosing the control points is named equal-length point (ELP). This method takes some points from the path Path($l_i$), and these points have equal length of path dist. The definition of equal-length point is described as

$$\begin{cases}P_i = \text{Path}(l_i) & = (X(l_i), Y(l_i), Z(l_i)) \\ & \wedge |l_i - l_{i-1}| = \text{dist}, i \in 2,3\ldots \\ P_1 = \text{Path}(l_1) & = (X(l_1), Y(l_1), Z(l_1)) \wedge l_1 = 0.\end{cases} \tag{24}$$

The length dist is longer, and the number of equal-length point is fewer.

*3) Equidistant Point:* The third method of choosing the control points is named equidistant point (EDP). This method takes some points from the path Path($l_i$), and these points have equal distance dist in 3-D space. The definition of equidistant point is described as

$$\begin{cases}P_i = \text{Path}(l_i) = (X(l_i), Y(l_i), Z(l_i)) \\ \quad \wedge \text{Euclid}(P_i, P_{i-1}) = \text{dist}, i \in 2,3\ldots \\ P_1 = \text{Path}(l_1) = (X(l_1), Y(l_1), Z(l_1)) \wedge l_1 = 0\end{cases} \tag{25}$$

where Euclid() is an operator of Euclid distance, and dist is the distance in 3-D space. Every two adjacent points have the same distance.

The performance of the three methods are different. By analysis (23)–(25), the performance of EDP is better than that of ELP, and the performance of ELP is better than that of SGP.

In the proposed approach, the robot motion planning is integrated with the path planning, thus the robots start to move once the overall task assignment. The robot navigation has the desired number of robots, even under uncertainties such as when some robots break down. Then proposed approach is capable of dealing with real-time and changing environment. The complexity of the optimization in one iteration is determined by the number of robots and the number of operational control points. During the whole process of path planning, the number of the control points using SGP method is the number of segmental points, while the number of control points using the other two methods is $\lceil L/\text{dist} \rceil$. Therefore, the complexity of the whole optimization process is $O(n)$, where $n$ is the number of robots. These optimization methods fit the online requirements of swarm robot system in real-time and dynamic environments.

## III. Simulation Result and Analysis

In order to demonstrate the effectiveness of the proposed algorithm for task assignment of swarm robots in 3-D dynamic environments, four different cases are studied in this section, including two cases for the comparison with the original algorithm, one case with arbitrary number of robots and targets with unexpected events such as some robots breaking down, and one case in a dynamic environment. For every case, the 3-D workspace is $100 \times 100 \times 100$. The algorithms are coded in MATLAB and implemented on a PC with an Intel Q8400 CPU, 4 GB RAM, and Win 7 operating system.

### *A. Comparison Studies*

To demonstrate the effectiveness of the proposed algorithm, a comparison study of original algorithm published for 2-D environments in [14] and [15] is conducted in this section with two cases.

In the first case, the initial positions of robots and targets are allocated randomly in the $100 \times 100 \times 100$ space. The number of robots is $n = 20$. The number of targets is $m = 20$. For the improved algorithm proposed in this paper, the initial learning rate of the parameter adjustment is $\eta = 0.5$ and the convergence rate is $\varsigma = 0.1$. The comparison results are shown in Fig. 5. Fig. 5(a) shows the results by using the original algorithm. Not all of the 20 robots can arrive successfully to the positions of the targets, while the path is unsmooth and the number of iterations is above 5000. Fig. 5(b) shows the results by using the improved algorithm optimization of adjustment parameters (OAPs). All of the targets are caught by robots under 257 iterations. The improved algorithm is better than the original one, but the paths are unsmooth. Fig. 5(c) shows the results by using the improved algorithm, which means using OAP and OBC, proposed in this paper. All of the targets are caught by robots with smooth and steady paths, while the number of iterations is just 257.

In the second case, the initial positions of the robots and targets are extreme. Some robots are surrounded by several targets, while some robots are far away from any target. Fig. 6(a) shows this situation. In this case, the number of robots is $n = 6$, the number of targets is $m = 6$, but Robot $R_1$ is surrounded by five targets, while Robot $R_2$ is far away from any target. As shown in Fig. 6(b), by using the original algorithm, Robot $R_1$ is hovering among the surrounded targets while Robot $R_2$ is not moving a step, which makes the swarm of robots unable to converge even up to 5000 iterations. By using the improved algorithm proposed in this paper, the swarm of robots can catch all the targets with smooth paths as shown in Fig. 6(b).

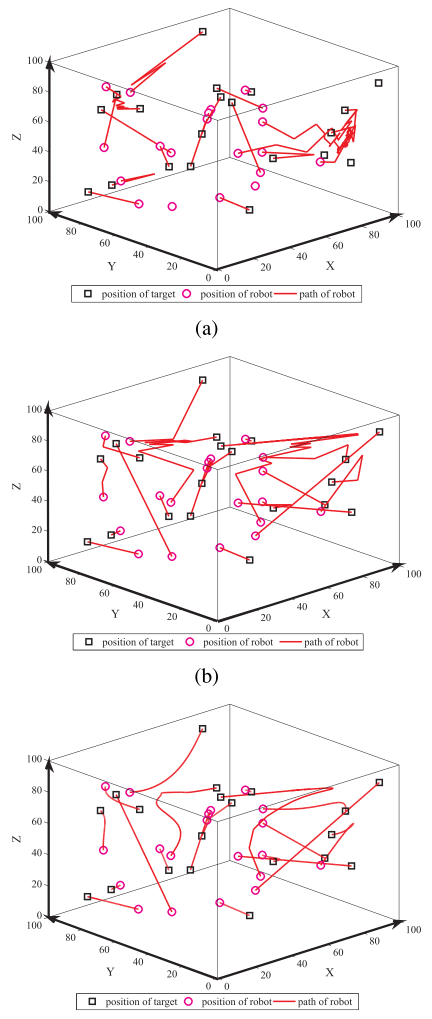
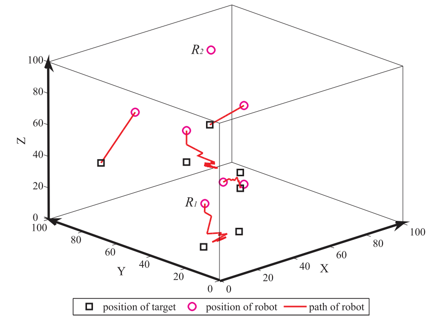


Fig. 5. Comparative results of path planning algorithms. (a) Using the original algorithm without optimization. (b) Using it with OAP. (c) Using it with OAP and OBC.

### *B. Arbitrary Number of Robots and Targets*

The proposed algorithm is robust and fault-tolerant. It can work in cases with arbitrary number of robots, targets and unexpected events with satisfaction. It allows for a sudden change of the environment such as a breakdown of some robots during the movement. Fig. 7 shows this kind of situation. There are 20 robots and 15 targets randomly distributed in the $100 \times 100 \times 100$ space. Assume that each robot has certain breakdown probability during the task process. It is set as 0.02 in this case. When a robot breaks down, it will stop and cannot execute the task anymore. This means other robots have to replace this broken robot to finish its task.

The simulation result is shown in Fig. 7, where the squares indicate the positions of the targets, the circles represents the initial positions of the robots, the real red lines are the paths of the robots, the asterisks indicate the positions of the broken robots, and the triangles indicate the positions of the nonstarted robots because of redundancy. These results show that all of the targets are caught by robots with smooth paths even when some robots are broken down during the process.


(a)

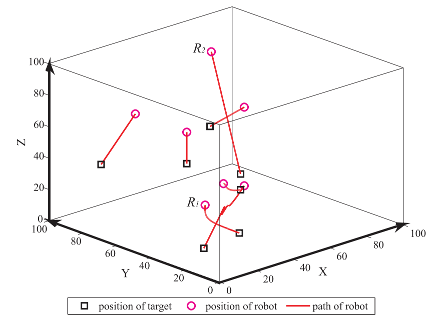

(b)

Fig. 6. Comparative results of path planning algorithms in extreme situations. (a) Using the original algorithm without optimization. (b) Using the proposed algorithm with optimizations.

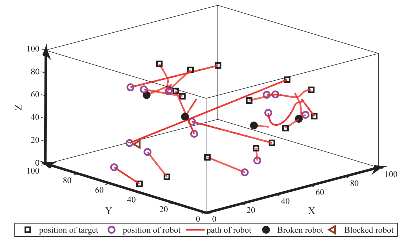


Fig. 7. Experiment of reliability.

If the number of targets is higher than the number of robots, these robots also can reach the targets one by one. From the mathematical perspective, it is easy to achieve. This is one of the advantages of this algorithm. According to this algorithm, especially under unpredictable and dynamic environments, any change in the number of robots would not affect task completion. When a robot reaches any target, it will become free again. This free robot will immediately enter the competition of catching other targets.

### C. In Dynamic Environment

The improved algorithm proposed in this paper can work properly in dynamic environments. In this case, assume that each target can move randomly in the workspace. Robots can still catch the target individually. Fig. 8 shows the results, where the squares represent initial positions of the targets and the asterisks depict the positions of the caught targets.

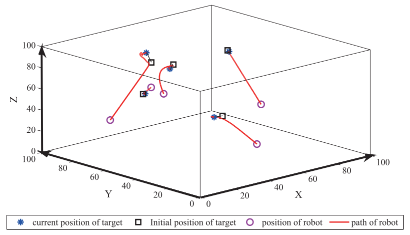


Fig. 8. Experiment of dynamic environment.

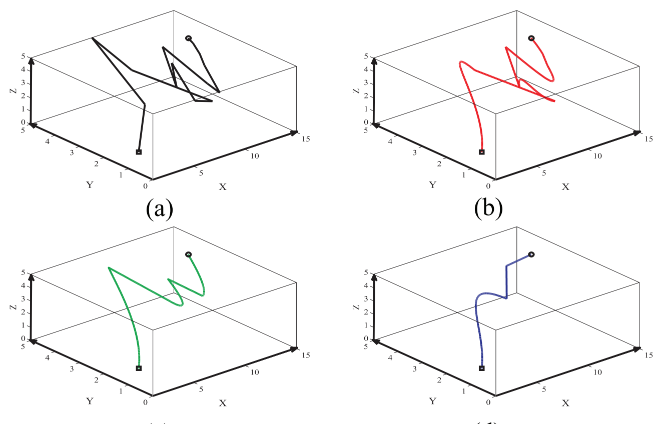

(a) (b) (c) (d)

Fig. 9. Result of Bezier optimization using SGP, ELP, and EDP. (a) Original path. Path by (b) SGP, (c) ELP, and (d) EDP.

## IV. Further Discussion

In this section, a further discussion on simulation results about the path optimization and parameter adjustment is presented in detail, including Bezier optimization, parameter adjustment, and the online ability of the path planning.

### A. Bezier Optimization

Bezier optimization methods are discussed in this section with simulation results in Fig. 9. An original path generated by the original method is illustrated in Fig. 9(a), where the starting point (marked as a circle) of the path is (2, 1, 1), and the destination (marked as a square) is (14, 4, 4). This means that a robot is marching from point (2, 1, 1) to point (14, 4, 4). Based on the same situation, three different methods of choosing the control points are applied to optimize the path as shown in Fig. 9(b)–(d). These three different methods named SGP, ELP, and EDP, described in Section II-E. It is obvious from this figure that the EDP method is the best one of the three optimization methods.

In comparison of these three methods theoretically, four evaluation items are introduced, including average angular velocity, total length of path, maximum angular velocity, and deviation of the path. Details are described as follows.

1) *Average Angular Velocity:* It is described as

$$\mathrm{AVG} = \frac{\int_0^L |\omega(l)| dl}{\int_0^L dl} \tag{26}$$

where $\omega(l)$ is the angular velocity when the robot marches to $l$ along the path $\mathrm{Path}(l)$. This value AVG

TABLE I
COMPARISON RESULTS BY DIFFERENT PARAMETERS

| Case | | MAX | | AVG | | Length | Deviation |
|---|---|---|---|---|---|---|---|
| | | $\alpha$ | $\beta$ | $\alpha$ | $\beta$ | | |
| Unopt | | 1.3661 | 1.9011 | 0.0119 | 0.0341 | 26.8601 | 0 |
| $Dist = 3 Num = 4$ | SGP | 2.2742 | 2.2623 | 0.0283 | 0.0315 | 22.5702 | 0.7662 |
| | ELP | 0.5008 | 1.8987 | 0.0100 | 0.0216 | 18.0781 | 1.2050 |
| | EDP | 0.7380 | 0.8393 | 0.0050 | 0.0058 | 13.5716 | 1.8877 |
| $Dist = 4 Num = 4$ | SGP | 2.2742 | 2.2623 | 0.0283 | 0.0315 | 22.5702 | 0.7662 |
| | ELP | 0.6245 | 0.7590 | 0.0074 | 0.0132 | 14.8374 | 1.6302 |
| | EDP | 0.6145 | 0.4788 | 0.0034 | 0.0029 | 13.5417 | 1.7032 |
| $Dist = 3 Num = 5$ | SGP | 1.6838 | 2.2147 | 0.0196 | 0.0279 | 20.1262 | 0.8605 |
| | ELP | 1.3851 | 1.6258 | 0.0116 | 0.0176 | 16.4481 | 1.3135 |
| | EDP | 0.1788 | 0.2770 | 0.0026 | 0.0029 | 13.1350 | 1.9952 |
| $Dist = 3 Num = 5$ | SGP | 1.6838 | 2.2147 | 0.0196 | 0.0279 | 20.1262 | 0.8605 |
| | ELP | 0.8278 | 1.9375 | 0.0067 | 0.0166 | 14.7790 | 1.5641 |
| | EDP | 0.0141 | 0.0152 | 0.0027 | 0.0027 | 13.0573 | 1.7380 |

is the major one of the evaluation items used to analyze the complexity of robotic control.

2) *Total Length of Path:* It is described as

$$\text{Length} = \int_0^L dl \tag{27}$$

where $L$ is the total length of the path $\text{Path}(l)$.

3) *Maximum Angular Velocity:* It is described as

$$\text{MAX} = \max(|\omega(l)|), 0 \leq l \leq L \tag{28}$$

where max() is the maximum operator. This value MAX is a minor one of the evaluation items in the analysis of the complexity of robotic control.

4) *Deviation of the Path:* It is described as

$$\text{Dev} = \frac{\int_0^L \text{Euclid}\left(\text{Path}_{\text{opt}}\left(\frac{l \times L_{\text{opt}}}{L}\right), \text{Path}(l)\right) dl}{\int_0^L dl} \tag{29}$$

where $L_{\text{opt}}$ is the total length of the optimized path, and $L$ is the total length of the original un-optimized path. This value analyzes the difference between the optimized path and the original un-optimized path.

Several experiments are conducted using different parameters. Table I summarizes the results in detail.

As shown in Table I, the complexity of robotic control is decreased after the optimizing process by Bezier curving. Meanwhile, three different methods (named SGP, ELP, and EDP) of choosing control point, have different performance on optimization. The performance of all optimized paths is better than the original un-optimized one. EDP has the best performance among three methods, while SGP is the easiest method to realize. However, the higher both dist and num are, the lower both MAX and AVG are. Hence, the complexity of robotic control decreases and the deviation between un-optimized paths and optimized paths increases. In conclusion, the Bezier curving optimization can decrease the complexity of robotic control effectively and lead to better performance.

## B. Parameter Adjustment

To analyze the effectiveness of the parameter adjustment, 100 experiments with 20 randomly distributed robots and targets are conducted. Because all the iteration numbers using the original method are over 4000. Fig. 10 records only the iteration numbers using two different parameter adjustment methods with three different learning rate.

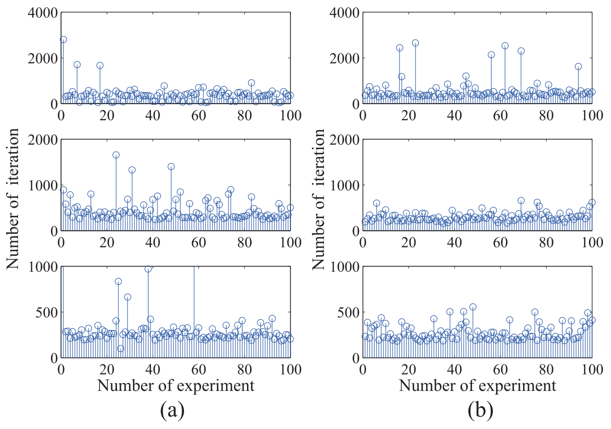


Fig. 10. Iteration number using two methods. (a) First method by (17). (b) Second method by (18).

In Fig. 10, the left and right column figures are the experimental results using two methods, respectively. These two methods are described in (17) and (18). The top, middle, and bottom figures are the results using the initial learning rate 0.4, 0.5, and 0.6, respectively.

As shown in Fig. 10, by adjusting parameters, the improved methods not only reduce two parameters in the original method to one, but also enable the number of iterations controllable and reduce the dependency and sensitivity of the parameters.

## C. Online Ability

To analyze the online processing ability, 100 experiments are carried out in this section. In these experiments, the number of robots is $n = 20$. The number of targets is $m = 20$. The initial positions of robots and targets are allocated randomly in the $100 \times 100 \times 100$ workspace. The initial learning rate is $\eta = 0.5$ using method (17). The convergence parameter is $\varsigma = 0.1$. Table II shows the statistical results extracted from the above 100 experiments with comparison of three different methods. One is the original method named SOM, and one is the improved method with only parameter adjustment named

TABLE II
STATISTICAL RESULT OF COMPARISON

| Case | SOM | | | SOM+OAP | | | SOM+OAP+OBC | | |
|---|---|---|---|---|---|---|---|---|---|
| | $MAX$ | $MIN$ | $AVG$ | $MAX$ | $MIN$ | $AVG$ | $MAX$ | $MIN$ | $AVG$ |
| Iterations | >5000 | 290 | >5000 | 1333 | 80 | 363 | 954 | 75 | 366 |
| Time | – | 0.1933 | – | 1.5765 | 0.1002 | 0.4344 | 4.9108 | 0.1660 | 1.5595 |

SOM+OAP, one is the improved method with parameter adjustment and Bezier curving named SOM+OAP+OBC.

As shown from Table II, the average iteration using the improved methods is much lower than that using original methods, and the average processing time using the improved methods is also less than that using original methods. The improved algorithm SOM+OAP decreases the number of iterations effectively. Even if the improved path planning algorithm SOM+OAP+OBC has two processes of optimization, the processing time is still low and less than the original method. This meets the requirements of real-time and online path planning in dynamic and complex environments.

## V. CONCLUSION

An improved path planning approach is proposed for task assignment of swarm robots in 3-D dynamic environments in this paper. With a self-organizing process, the proposed approach has some interesting features and advantages. It is suitable for online path planning with arbitrary number of robots and tasks. It is capable of dealing with sudden changes in the situation such as the breakdown of some mobile robots. It can deal with changing environments such as movable targets. It can generate steady and smooth paths and reduce the complexity of robotic control with strong fault tolerance and robustness. Compared with the original approach, the improved aspects are summarized as follows.

1) The improved approach extends to 3-D environments from the original 2-D environments.
2) In order to solve the misconvergence problem in the original approach, two optimizing methods of adjusting the learning rate are proposed, which enable all of the robots to converge in controllable iterations.
3) Bezier path optimization methods are applied to achieve smooth and steady paths with reduction of the complexity of robotic control.

In this paper, the cooperation and coordination among a large number of robots and target positions in dynamic 3-D environments are mainly considered, while obstacle avoidance situation is not considered. In our earlier study [34], obstacle avoidance situation is considered in 2-D environments. For further work, it would be taken into account in detail.

## REFERENCES

[1] G. Bekey and J. Yuh, "The status of robotics," *Robot. Autom. Mag.*, vol. 15, no. 1, pp. 80–86, Mar. 2008.

[2] C. Blum and D. Merkle, *Swarm Intelligence*. Berlin, Germany: Springer-Verlag, 2008.

[3] F. Aznar *et al.*, "Agents for swarm robotics: Architecture and implementation," in *Highlights in Practical Applications of Agents and Multiagent Systems*. Berlin, Germany: Springer-Verlag, 2011, pp. 117–124.

[4] R. Madhavan, "Robots in military and aerospace technologies [news and views]," *Robot. Autom. Mag.*, vol. 17, no. 2, p. 6, 2010.

[5] B. Bethke, M. Valenti, and J. P. How, "UAV task assignment," *IEEE Robot. Autom. Mag.*, vol. 15, no. 1, pp. 39–44, Mar. 2008.

[6] C. Luo, A. P. Espinosa, D. Pranantha, and A. De Gloria, "Multi-robot search and rescue team," in *Proc. IEEE Int. Symp. Safety Security Rescue Robot.*, Kyoto, Japan, 2011, pp. 296–301.

[7] K. Zhou and S. I. Roumeliotis, "Multirobot active target tracking with combinations of relative observations," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 678–695, Aug. 2011.

[8] H. Hatime, R. Pendse, and J. M. Watkins, "Comparative study of task allocation strategies in multirobot systems," *IEEE Sensors J.*, vol. 13, no. 1, pp. 253–262, Jan. 2013.

[9] L. Sabattini, C. Secchi, M. Cocetti, A. Levratti, and C. Fantuzzi, "Implementation of coordinated complex dynamic behaviors in multirobot systems," *IEEE Trans. Robot.*, vol. 31, no. 4, pp. 1018–1032, Aug. 2015.

[10] A. Yazici, G. Kirlik, O. Parlaktuna, and A. Sipahioglu, "A dynamic path planning approach for multirobot sensor-based coverage considering energy constraints," *IEEE Trans. Cybern.*, vol. 44, no. 3, pp. 305–314, Mar. 2014.

[11] A. Macwan, J. Vilela, G. Nejat, and B. Benhabib, "A multirobot path-planning strategy for autonomous wilderness search and rescue," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1784–1797, Sep. 2015.

[12] X. Yi, A. Zhu, and Z. Ming, "A bio-inspired approach to task assignment of multi-robots," in *Proc. IEEE Symp. Swarm Intell. (SIS)*, Orlando, FL, USA, Dec. 2014, pp. 1–5.

[13] A. Zhu and S. X. Yang, "A survey on intelligent interaction and cooperative control of multi-robot systems," in *Proc. IEEE Int. Conf. Control Autom.*, Xiamen, China, 2010, pp. 1812–1817.

[14] A. Zhu and S. X. Yang, "A neural network approach to dynamic task assignment of multirobots," *IEEE Trans. Neural Netw.*, vol. 17, no. 5, pp. 1278–1287, Sep. 2006.

[15] A. Zhu and S. X. Yang, "An improved SOM-based approach to dynamic task assignment of multi-robots," in *Proc. World Congr. Intell. Control Autom.*, Jinan, China, 2010, pp. 2168–2173.

[16] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, 2004.

[17] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proc. IEEE*, vol. 94, no. 7, pp. 1257–1270, Jul. 2006.

[18] B. P. Gerkey and M. J. Mataric, "Sold!: Auction methods for multirobot coordination," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 758–768, Oct. 2002.

[19] P. Brass, F. Cabrera-Mora, A. Gasparri, and J. Xiao, "Multirobot tree and graph exploration," *IEEE Trans. Robot.*, vol. 27, no. 4, pp. 707–717, Aug. 2011.

[20] Y. T. Kotb, S. S. Beauchemin, and J. L. Barron, "Workflow nets for multiagent cooperation," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 1, pp. 198–203, Jan. 2012.

[21] L. Zhong *et al.*, "A task assignment algorithm for multiple aerial vehicles to attack targets with dynamic values," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 236–248, Mar. 2013.

[22] A. Jevtic, A. Gutiérrez, D. Andina, and M. Jamshidi, "Distributed bees algorithm for task allocation in swarm of robots," *Syst. J.*, vol. 6, no. 2, pp. 296–304, Jun. 2012.

[23] D. Zhu, H. Huang, and S. X. Yang, "Dynamic task assignment and path planning of multi-AUV system based on an improved self-organizing map and velocity synthesis method in three-dimensional underwater workspace," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 504–514, Apr. 2013.

[24] J. Ni and S. X. Yang, "Bioinspired neural network for real-time cooperative hunting by multirobots in unknown environments," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2062–2077, Dec. 2011.

[25] H. Li, S. X. Yang, and M. L. Seto, "Neural-network-based path planning for a multirobot system with moving obstacles," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 39, no. 4, pp. 410–419, Jul. 2009.

[26] C. Luo and S. X. Yang, "A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments," *IEEE Trans. Neural Netw.*, vol. 19, no. 7, pp. 1279–1298, Jul. 2008.

[27] D. Ribas, N. Palomeras, P. Ridao, M. Carreras, and A. Mallios, "Girona 500 AUV: From survey to intervention," *IEEE/ASME Trans. Mechatronics*, vol. 17, no. 1, pp. 46–53, Feb. 2012.

[28] M. Tortonesi *et al.*, "Multiple-UAV coordination and communications in tactical edge networks," *IEEE Commun. Mag.*, vol. 50, no. 10, pp. 48–55, Oct. 2012.

[29] H. Chen, K. Chang, and C. S. Agate, "UAV path planning with tangent-plus-Lyapunov vector field guidance and obstacle avoidance," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 2, pp. 840–856, Apr. 2013.

[30] T. Ender *et al.*, "Systems-of-systems analysis of ballistic missile defense architecture effectiveness through surrogate modeling and simulation," *Syst. J.*, vol. 4, no. 2, pp. 156–166, Jun. 2010.

[31] T. Kohonen, *Self-Organization and Associative Memory*. Berlin, Germany: Springer-Verlag, 1987.

[32] T. Kohonen, "The self-organizing map," *Neurocomputing*, vol. 21, no. 1, pp. 1–6, 1998.

[33] T. Kohonen *et al.*, "Self-organization of a massive document collection," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 574–585, May 2000.

[34] A. Zhu and S. X. Yang, "A framework for coordination and navigation of multi-robot systems," in *Proc. IEEE Int. Conf. Autom. Logist.*, Hong Kong, 2010, pp. 350–355.

**Xin Yi** received the B.Sc. degree in computer science from Hunan International Economics University, Hunan, China, in 2011, and the master's degree from the School of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China, in 2014.

His current research interests include neural networks, machine learning, and multiagent systems.

**Anmin Zhu** (S'04–M'09) received the B.Sc. and M.Sc. degrees in computer science from Tongji University, Shanghai, China, in 1987 and 1990, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Guelph, Guelph, ON, Canada, in 2005.

He is currently a Faculty Member with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include fuzzy systems, neural networks, robotics, self-organizing arrangement, machine intelligence, and multiagent systems.

**Simon X. Yang** (S'97–M'99–SM'08) received the B.Sc. degree in engineering physics from Beijing University, Beijing, China, in 1987, the first M.Sc. degree in biophysics from the Chinese Academy of Sciences, Beijing, in 1990, the second M.Sc. degree in electrical engineering from the University of Houston, Houston, TX, USA, in 1996, and the Ph.D. degree in electrical and computer engineering from the University of Alberta, Edmonton, AB, Canada, in 1999.

In 1999, he joined the School of Engineering, University of Guelph, Guelph, ON, Canada, where he is currently a Professor and the Head of the Advanced Robotics and Intelligent Systems (ARIS) Laboratory. He has authored or co-authored about 400 refereed papers, including about 200 journal papers. His current research interests include intelligent systems, robotics, sensors and multi-sensor fusion, wireless sensor networks, control systems, soft computing, and computational neuroscience.

Prof. Yang serves as the Editor-in-Chief of *International Journal of Robotics and Automation* and the *Journal of Robotics and Artificial Intelligence*, as an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, PART B: CYBERNETICS, and several other journals, and as an Advisory Editorial Board Member of *Intelligent Automation and Soft Computing*, and other journals. He was the General Chair of the 2011 IEEE International Conference on Logistics and Automation and a recipient of the Distinguished Professor Award, University of Guelph.

**Chaomin Luo** (S'01–M'08) received the B.Sc. degree in radio engineering from Southeast University, Nanjing, China, in 1994, the M.Sc. degree in engineering systems and computing from the University of Guelph, Guelph, ON, Canada, in 2002, and the Ph.D. degree in electrical and computer engineering from the University of Waterloo, Waterloo, ON, Canada, in 2008.

In 2008, he was an Assistant Professor with National Taipei University, Taipei, China. In 2009, he joined the University of Detroit Mercy, Detroit, Michigan, USA, where he is currently an Associate Professor with Advanced Mobility Laboratory. His current research interests include robotics and automation, intelligent systems, computational intelligence, mechatronics, very large scale integration, and embedded systems.