Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage:www.elsevier.com/locate/robot

Self-Organized Swarm Robot for Target Search and Trapping Inspired by Bacterial Chemotaxis

Bin Yang [a,b,d], ***Yongsheng Ding*** [a,b *], ***Yaochu Jin*** [b,c], ***and Kuangrong Hao*** [a,b]

[a] *Engineering Research Center of Digitized Textile & Apparel Technology, Ministry of Education, Donghua University, Shanghai 201620, P. R. China*
[b] *College of Information Sciences and Technology, Donghua University, Shanghai 201620, P. R. China*
[c] *Department of Computing, University of Surrey, Guildford, GU2 7XH, United Kingdom*
[d] *Department of Mathematics, Huaiyin Normal University, Huai'an, Jiangsu, 223300, P. R. China*

ARTICLE INFO

Article history:
Received 00 December 00
Received in revised form 00 January 00
Accepted 00 February 00

Keywords:
swarm robots
distributed control
target searching
target trapping
bacteria chemotaxis
self-organization

ABSTRACT

Target search and trapping using self-organized swarm robots have received increasing attention in recent years but control design of these systems remains a challenge. In this paper, we propose a decentralized control algorithm of swarm robot for target search and trapping inspired by bacteria chemotaxis. First, a local coordinate system is established according to the initial positions of the robots in the target area. Then the target area is divided into Voronoi cells. After the initialization, swarm robots start performing target search and trapping missions driven by the proposed bacteria chemotaxis algorithm under the guidance of the gradient information defined by the target. Simulation results demonstrate the effectiveness of the algorithm and its robustness to unexpected robot failure. Compared with other commonly used methods for distributed control of swarm robots, our simulation results indicate that the bacteria chemotaxis algorithm exhibits less vulnerability to local optimum, and high computational efficiency.

1. INTRODUCTION

Swarm robots [1], also known as multi-robots systems, include a group of simple physical robots. Swarm robotic systems are believed to be more reliable or effective in accomplishing tasks than single robots. Target search and trapping in a given area is a frequently encountered task, such as terrorist search, pollution detection, disaster area monitoring and rescuing. Therefore, target search and trapping using self-organized swarm robots have received increasing attention in recent years, although design of self-organized controller of such systems remains a grand challenge.

Target search aims to detect targets in a given area, which is also known as area coverage. Area coverage problem has been tackled for years using either one single robot [2][3] or multi-robots/swarm robots [4][5]. It is also indispensable in simultaneous localization and mapping. The difference between area coverage and mapping is that the former does not need the precise location of the targets in the environment.

* *Corresponding author.* Tel.:+ 86-21-6779-2323; fax: +86-21-67792353.
E-mail address: ysding @ dhu.edu.cn

Target trapping is to surround the detected targets using robots in a certain pattern. Target trapping methods are similar to pattern formation using swarm robots, which is often concerned with the formation of a complex shape. In recent years, much work has been devoted to solving complex shape formation using swarm robots, which can be divided into global shape information [6][7] and local affection [8][9]. Feng et al. developed a finite-time formation control framework that included global information and local information [6]. Meng et al. proposed a morphogenetic approach to swarm robotic system using a gene regulatory network (GRN) to form complex shapes. The shape for the GRN is described by Non-Uniform Rational B-Splines in initialization [7]. Ghods et al. designed a control algorithm based on the heat partial differential equation (PDE) and extremum seeking for deploying a group of agents [8]. Ji et al. used the leader-follower structure with the neighbor-based rule determined by the topology structure of an interconnection graph to solve the formation control problem [9].

The main challenge in swarm robot target search and trapping is the design of a distributed control and coordination mechanism. Much research has been conducted on designing the distributed algorithms, which can be roughly classified into four categories, i.e. virtual structure algorithms, leader-following algorithms, artificial potential field algorithms, and bio-inspired algorithms.

Virtual structure algorithm was firstly proposed late 1990s [10][11]. This method considers a robot as a single virtual rigid structure and adopts traditional control algorithms designed for single robots/agents.

Leader-following (virtual leader) algorithms have attracted huge interest since they were proposed [12][13][14][15][16]. Typically, an algorithm is designed to control the leader robot, while others (followers) are located by the relative positions with their neighbors.

Artificial potential field (APF) algorithms were proposed for real-time path planning for robots to solve obstacle avoidance problem, and became the most widely studied distributed control methods. In the APF methods, it is assumed that the robots combine attraction with the goal, and repulsion from obstacles. Much work has been reported to adapt the APF algorithm to controlling swarm robots [17][18]. However, the APF algorithm is known to easily get trapped in a local minimum. Therefore, many ideas for addressing this problem have been proposed.

Bio-inspired algorithms for self-organized swarm robot to target search and trapping have been proposed in recent years inspired by a variety of biological systems, such as flocking behaviors in social insects. For examples, Garcia et al. proposed a simple ant colony optimization meta-heuristic (SACOdm) algorithm to solve the path planning problem of autonomous mobile robots [19]. The SACOdm methods determine the robots' path based on the distance from the source to the target nodes, where the ants remember the visited nodes. Zhu et al. proposed an improved scout ant algorithm to achieve the optimal static navigation path for swarm robots [20]. They also proposed a target interception algorithm called the multi-scout ants' cooperation (MSAC) for robots based on sub-goal forecasting and a scout ant algorithm [21]. The scout ants search randomly the targets without the influence of pheromone. The MSAC algorithm uses two families of ants, one searching the current position and the other from the target, which is therefore a bidirectional search. All the visited grids by the scout ants of the two families are stored in a global table to avoid repeated search. This method has been shown to have greatly improved the search efficiency and can be implemented for real-time path planning. The robotic Darwinian PSO (RDPSO) algorithm, proposed by Couceiro et al., is inspired from the natural selection and social exclusion [22][23][24]. The RDPSO algorithm evaluates the dynamical partitioning of all the robots and exchanges the information needed to adjust its parameters based on a number of context-based evaluation metrics. Robots are divided into multiple swarms and the robots in a swarm can communicate with each other. The main advantage of the RDPSO algorithm is that it can escape from local optimal solutions and reduce the needed information exchange.

The bacterial foraging optimization (BFO) algorithm was proposed to mimic the *E. coli* bacteria foraging behaviours in the intestine [25]. Bacteria are one of the simplest species living on the earth. Many bacterial species are single-cells, and are associated with each other in their characteristic patterns. They can, e.g., sense the nutrient information around them and swim to places having a higher nutrient concentration. The BFO algorithm has become very popular [26][27] due to its strong ability in escaping from local optima and faster convergence compared with other heuristic methods. Ideas for improving the BFO algorithm have also been suggested. For example, Müller et al. proposed a bacterial chemotaxis (BC) algorithm based on a model of bacterial chemotaxis [28]. Bacteria chemotaxis is the phenomenon that bacteria direct their movements by following the chemical concentration gradient around them. For example, bacteria can find food by swimming towards area with a higher concentration of nutrition.

Bio-inspired algorithms possess the merits of expansibility and flexibility for the self-organized control of swarm robots. However, some of these algorithms need strict constraints such as absolute initial positions [19]. In addition, these heuristic algorithms often need to perform large amount of calculation, incurring prohibitive computational cost [10][19][22][23]. Therefore, how to improve the control efficiency, such as reducing the convergence time becomes a primary concern in these methods.

In this paper, we propose a self-organization algorithm for swarm robot target search and trapping inspired by bacteria chemotaxis. The advantages of the proposed algorithm are: 1) Robots are less likely to get trapped in local optimums; 2) The computational cost is low; 3) Predefined global coordinate is not needed for accomplishing the tasks.

The rest of this paper is organized as follows: Section 2 introduces the framework and the basic assumptions used in the BC algorithm. Section 3 describes the proposed BC algorithm for swarm robot target search and trapping. Comparative simulation studies are presented in Section 4. Section 5 concludes the paper and discusses future work.

2. The Framework and Basic Assumptions

2.1. The Framework

The proposed framework for target search and trapping problem using swarm robots can be divided into the following steps, as also shown in Fig. 1.

Step 1. Establish a local coordinate system according to the robots' initial positions in the target area.

Step 2. Divide the area to be searched into N (the number of robots) cells using a Voronoi diagram.

Step 3. Use the BC algorithm to search the target in each Voronoi cell.

Step 4. Surround the detected targets using the BC algorithm.

Some definitions in the above steps are elaborated as follows.

1) Construction of a local coordinate system

At first, we initialize the position of the robots using a local coordinate system constructed based on their relative positions [29]. A reference robot is selected first, whose position is set to be the origin of the local coordinate system. Then, the position of other robots can be determined based on their relative distance to the reference robot. The same can be done to determine the position of any point in the area using the local coordinate system.

2) Voronoi diagram

Voronoi diagram [30] is a mathematical method to divide space into a number of regions. A number of points (sometimes called seeds, sites, or generators) are given at first. Each point has a corresponding region that consists of all the points closer to it than to any others. The regions are called Voronoi cells.

To improve the efficiency of target search by swarm robots, Voronoi diagram is used here to divide the space into a number of small regions, termed cells. Each robot detects the target in the i-th cell using the BC algorithm.

3) Moving state

We define robot i at time instant j as $R_{i,j}$ with two states for area coverage and trapping, respectively (refer to Fig.1). The two states are:

a) Searching state. After initialization, the robots are all in a searching state when the area has not been covered. They aim to detect targets within its own Voronoi cells and then search in others' after covering their own. The details about the algorithm for the searching state using the BC algorithm will be presented later on.

b) Trapping/Pattern formation state. The robots will change into the trapping/pattern formation state if the area has been covered. We set a high concentration to the positions for trapping the detected targets. All robots switch to the trapping/pattern formation state and trap the targets driven by a gradient.

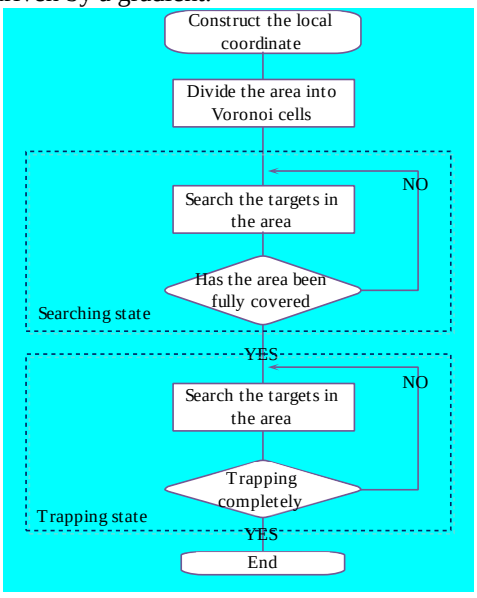


Fig. 1 The flowchat for target searching and trapping by swarm robots.

2.2. Assumptions

To develop control algorithms for target search and trapping in a given area by using swarm robots, we make the following assumptions:

1) All targets are detectable

The main problem to be solved in this paper is to detect the targets in a certain area and surround them. Therefore, we assume that the target can be detected if it is within the sensing range of the robot.

2) All position information can be shared among the robots

Here we assume that robots are able to self-localize them in the given area using on-board equipment, such as blueteeth [31] or WIFI [32]. We also assume that the robots' communication range can cover the whole area. Thus, the robots can know other robots' positions, and do not need the whole field information.

3) The area is 2-dimensional and bounded

We assume that the edges of the area to be searched are known for develop the control algorithm. In this paper, we consider solving the problems in a continuous 2-dimensional area of a square-shape. The robot can visit all positions in the area except that the position is occupied by an obstacle or other robots.

3. A Bacteria Chemotaxis Algorithm for Target Search and Trapping

The BC algorithm developed in this work is adapted from the classical bacteria foraging optimization (BFO) algorithm proposed by Passino [27][33], which is used to solve optimization problems.

The main idea of the BC algorithm is to consider every

robot as a bacterium. Then we can use the mechanism of bacteria chemotaxis to solve distributed control problems of swarm robots. The bacteria aim to find a position with a higher level of concentration of nutrient. To this end, similar chemotaxis mechanisms in the BFO algorithm can be adopted. During the search for a higher concentration of nutrients, the targets that can be sensed by robots will be detected.

The main ideas of the BC algorithm are as follows:

1) Set the concentration of the targets' position to a large value *A*, where the grids are colored in black as illustrated in Fig. 2.

2) Calculate the gradient for the robots' chemotaxis direction δ.

3) Bacteria swim along the direction of δ to approach the target position.

The nutrient concentration will decrease once it is covered by a bacterium. We find that this contributes to the BC algorithm's ability to get out from local optimums because of gradient change caused by the concentration decrease. This has also been demonstrated in our simulations.

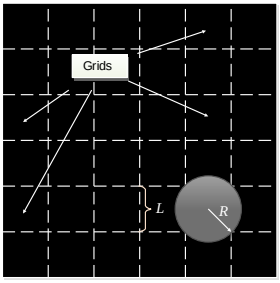


Fig. 2 Grids selection on the area

In the following, we present the details of the BC algorithm for target search and trapping, respectively.

3.1 The BC Algorithm in Searching State

At first, the position of the robots and the concentration of each cell are initialized. An example is given in Fig. 3, where 10 robots are randomly initialized in the area, where

$$C = A_0 \tag{1}$$

where A_0 is a large constant denoting the initial concentration. We set the step length of robot *i* equals to its real-time velocity V_i, $i = 1,2,...n$. Then the given area for target search is partitioned into 2D grids as illustrated in Fig. 2. Let $L = R * \sqrt{2}$, where *L* is the length of grids, *R* is the robots' sensing range. Therefore, *C* is a matrix whose dimensions can be calculated by the size of the area and the length of the grids. When each robot has reached all of the grids in its cell, the robot is considered to have covered the cell. After initialization, robot *i* moves in the direction to the nearest grid with the concentration of A_0. If the robot has more than one nearest cells to search, we calculate the direction δ to the next target using the method described as follows, referring to Fig. 4.

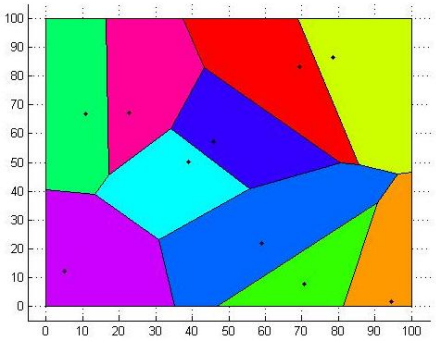


Fig. 3 Ten robots (black points) and their Voronoi cells.

The chemotaxis step can be expressed by

$$\frac{dN_i(t)}{dt} = b\delta + aD_i \tag{2}$$

where $N_i(t) = [N_{i,x}(t); N_{i,y}(t)]$ is the position of the *i*-th robot's position, $\delta = [\delta_x; \delta_y]$ is the direction, *a* is the constant coefficient of D_i, *b* is the constant coefficient determined by the robots' speed. $D_i = [D_{i,x}; D_{i,y}]$ is the summed interaction between the *i*-th robot and other robots to avoid collision between robots, which can be calculated by

$$D_i = \sum_{m=1}^{B_i} D_i^m \tag{3}$$

where B_i is the number of neighbors for robot *i*, and $D_i^m = [D_{i,x}^m; D_{i,y}^m]$ is the interaction between robot *m* and robot *i* which can be given by

$$D_i^m = \frac{(N_i - N_m)}{\| N_i - N_m \|} \tag{4}$$

As the energy consumption of robots to move straightforward is less than that for robots to make turns, the robots will not change their directions unless they are trapped in a local optimum. As shown in Fig. 4, to escape from a local optimum, the directions of the robots are determined as follows:

$$\begin{aligned} P(X = \delta) = {} & P(\nu = 1)P(X = \delta, \nu = 1) \\ & + P(\nu = 0)P(X = \delta, \nu = 0) \end{aligned} \tag{5}$$

where ν is denoted as 1 or 0 when the robot moves straightforward or makes a turn, respectively. Set $\theta = arctan(\delta_y / \delta_x)$, angle δ yields a probability density distribution $P(X = \delta, \nu = 0) \sim N(\mu, \sigma^2)$. This means the robot always has a higher probability to choose a smaller

$|\theta|$ as shown in Fig. 4. When robots are moving straightforward,

$$\begin{cases} P(v=1)=1 \\ P(v=0)=0 \end{cases} \quad (6)$$

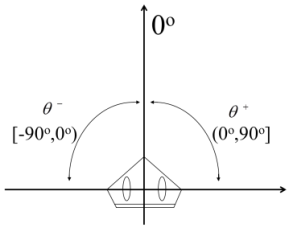


Fig.4. Direction setting for robots' movement

After the direction was selected, the robots keep moving driven by the BC algorithm for area coverage. Let $(Px_{i,j}, Py_{i,j})$ represents the position of the *i*-th robot at the *j*-th chemotaxis step. In the progress, the concentrations in the area covered by the sensors of the robots are

$$C(Px_{i,j}, Py_{i,j}) = C(Px_{i,j}, Py_{i,j})/2 \quad (7)$$

We set the initial concentration C_i in the whole area using Eq. (1). Once this initialization has been done, the robots will switch into the searching state. When the robots are moving in the area, the concentration of the positions that can be covered by the robot' sensors are updated using Eq. (7). After the robots have been initialized and the filed has been divided into a number of Voronoi cells, the virtual nutrient matrix of the *i*-th Voronoi cell for the *i*-th robot can be described as M_i in Algorithm 1. This means that each robot's movement is guided with the nutrient concentration as recorded in M_i. In this way, each robot R_i can detect the targets located in the area covered by the *i*-th Voronoi cell.

Once a robot has covered its own cell and found the targets, it will switch into the leading state as a leader. When other robots have covered their own Voronoi cells, they move towards the cells where the concentration equals A_0 outside their own cells.

The pseudo code of the proposed BC algorithm for swarm robot target search is presented in Algorithm 1.

3.2 The BC Algorithm in Trapping state

The BC algorithm can now be used to solve the target trapping problem by robots in the trapping state.

Once a robot detects a target, it communicates and shares the target's position with others. Consequently, we can set the concentration of the target position to a large constant. When the robots turn to the leading state, robot *i* finds its target $T(i)$ using the BC algorithm as described in Algorithm 2. Then the following robots, which have detected no target, will swim to $T(i)$ according to Eq. (2).

ALGORITHM 1. BC ALGORITHM FOR TARGET SEARCH

1: Construct a local coordinate system by selecting one reference point and using the relative position between the robots, define the range of the area, initialize concentration *C*. Divide the area into Voronoi cells according to the positions of the robots. Get the total number of the grids, S_i , in the *i*-th cell. $\mathbf{C}_i$ and M_i represent the grids' concentrations and the grids in the *i*-th Voronoi cell, respectively.
2: For the *i*-th robot R_i in M_i, chemotaxis loop: $j=j+1$
3: Set concentration of each grids $\boldsymbol{G}_i(j)$ covered by S_i as
$C(Px_{i,j}, Py_{i,j}) = C(Px_{i,j}, Py_{i,j})/2$
4: Find the grids $\mathbf{G}_i(j)$ where the nutrient concentration is the highest and calculate the possible direction for next chemotaxis
5: If $\delta_i(j) \in \boldsymbol{\delta}_i(j+1)$ & $|P_i(j+1) - P_i(j)| < \dot{o}$ & $max(\mathbf{C}_i) = A_0$
$\delta_i(j+1) = \delta_i(j)$ //moving straightforward
6: Else $P(X=\delta_i) = P(X=\delta_i, v=0), \delta_i \neq 0$
//trapped in a local optimum
7: End
8: $\frac{dN_i}{dt} = b\delta + aD_i^m$
9: If $max(\mathbf{C}_i) = A_0$ //the *i* cell is not fully covered
Goto Step 2
10: Else $M_i = M(Px, Py),\ where\ C(Px, Py) = A_0$
//search in other cells
11: End
12: End

ALGORITHM 2. BC ALGORITHM FOR TARGET TRAPPING

1: If a robot is in the trapping state and has found nr targets in the area represented by $T = \{T_1, T_2 ... T_{nr}\}$, the distance from each target to robots *i* is $Dt(i,k) = \| R_i - T_k \|$, $k \in (1,2,...nr)$, nr is the number of the targets.
2: Set the concentration of the points (boundary) at a distance of Rt from each target as $C(T_i) = A_0$.
3: In order to minimize the trajectory and trap the targets evenly, each target will be trapped by $nt = \lfloor (n/nr) \rfloor$ robots. At first, define target matrix for robot *i*
$T(i) = 0, i \in [1,n]$
Then find the nearest target for robot *i*.
for l=1:nr // *for each target*
$tm = \min(Dt(T==0, l), nr)$ //select the *nr* nearest robots for the *l*-th target
$T(tm) = l$
End
4: The direction robot *i* can be calculated by
$\delta_i(j+1) = grad(T(i))$

5: $\frac{dN_i}{dt} = b\delta + aD_i^j$

6: End

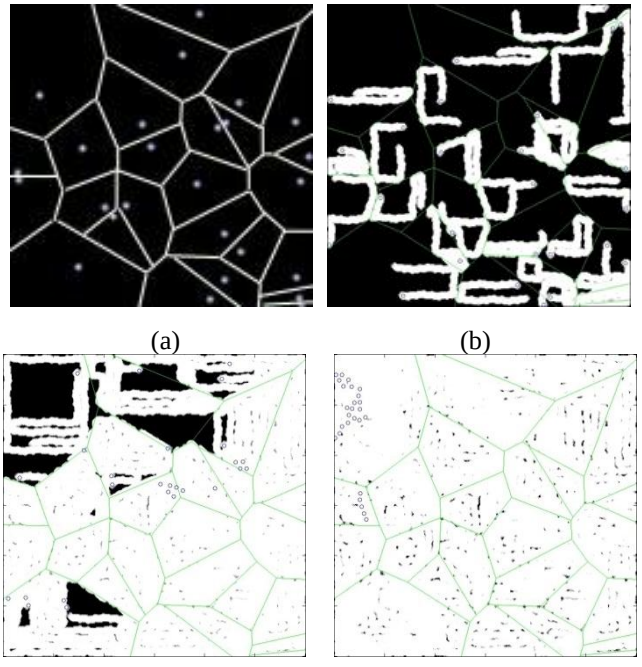
4. Simulation Results and Analysis

To evaluate the proposed BC algorithm for target search and trapping, we have performed some empirical simulation studies. Here we initialize the robots in a 300×300 square field, set the coefficient a=0.025 and the sensing range to 0.8.

4.1. Target Search

In this section, it will be shown that how swarm robots accomplish target search based on the proposed BC algorithm. Fig. 5 shows a set of snapshots of the searching process by 26 swarm robots controlled by the proposed BC algorithm. Fig. 5(a) shows 26 robots randomly located in a 300×300 square area. The reference robots are selected from the edge of convex region formed by the swarm robots and a local coordinate system is constructed.

Based on the local coordinate system, the initial position of robots can be obtained and establishment of Voronoi diagram can be carried out, as illustrated in Fig. 5 (a). Fig. 5 (b) and Fig. 5(c) show the robots covering the area using the BC algorithm. Fig.5 (d) indicates that the swarm robots that have covered the whole area. From the snapshots we can find that there are many grey areas, which indicate that the concentration in these areas is still very high.



(a) (b)

(c) (d)

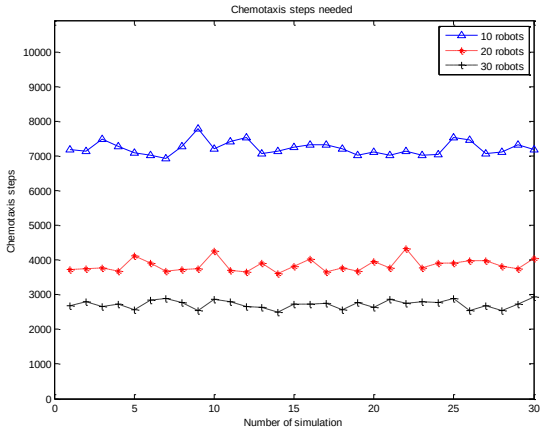
Fig. 5 Area coverage by 26 robots based on Voronoi diagram and the BC algorithm.

By contrast, the locations in white are those that have been searched repeatedly by the robots' sensors. This indicates there is a waste of energy and the search is inefficient.

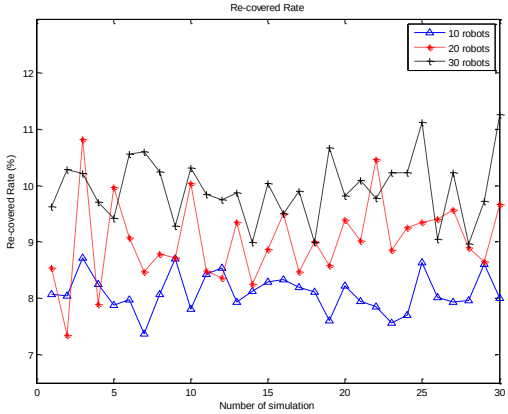
Table I lists the mean and standard deviation of the chemotaxis steps of the swarm robots and the number of re-covered grids. For each case where 10, 20 and 30 swarm robots are used, respectively, 30 independent runs have been conducted.

TABLE I
AREA COVERAGE PERFORMANCE USING THE BC ALGORITHM

No. of robots	Chemotaxis steps	Re-covered rate (%) (Re-covered / all grids)
10	7009±214	8.0970±0.3364
20	3839±181	9.0333±0.7325
30	2638±118	9.9408±0.5846



(a) Chemotaxis steps



(b) Recovered rate (%)

Fig. 6 Area coverage by swarm robots with the BC algorithm.

From Table I and Fig. 6, we can find that when 10, 20, 30 robots are used, they can cover the area in almost 7009, 3839, 2638 steps, respectively. At the same time, the re-covered rates of the robots are about 8.1%, 9.0% and

9.9%. This indicates that if we use more robots to cover the area, a smaller number of chemotaxis steps will be needed. These results show the BC algorithm can efficiently cover the area within certain steps and re-cover the grids. In other words, the more robots are used, the quicker it is to cover the area. However, the re-covered area will increase at the same time.

In the following description, the ability of the swarm robot systems to avoid obstacles is illustrated. To this end, we consider a scenario with six stationary obstacles and 26 robots in the area. Fig. 7 shows a number of snapshots observed in this simulation. From Fig. 7, especially in the area marked with around rectangle, we can see that when a robot detects an obstacle, it can keep away from the obstacle to avoid collision.

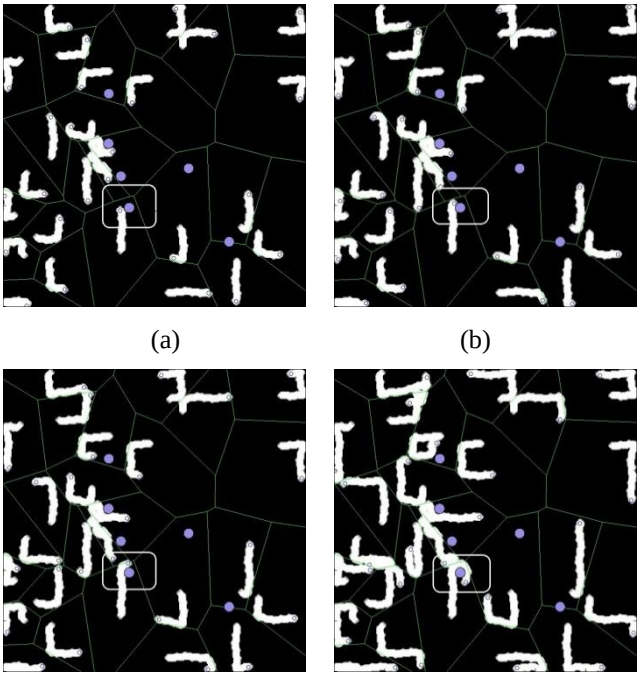


Fig. 7 Six stationary obstacles that can be avoided by 26 robots. It is described in fig.(a) and fig.(b) that the robot marked in the rounded rectangle detects an obstacle and the edge of the cell. In fig.(c) and fig.(d), the robot moves in the direction of avoiding the obstacle under the controlling of BC algorithm .

In order to further evaluate obstacle avoidance capability of the proposed BC algorithm, we consider another scenario in which an obstacle is moving across the area, as shown in Fig. 8.

In the area marked with a red rectangle at the right bottom in Fig. 8, we can find that the robot can avoid the obstacle if it detects the obstacle. In the simulation, the robot moves very close to the obstacle and a collision with the obstacle could have happened for physical robots. This is because the robot and the obstacle are moving in opposite directions. Therefore, the robot has moved backward for avoiding a collision with the obstacle.

Fig. 8 (a) illustrates a situation where the robot could have got trapped in a local minimum. The force diagram for the robot is illustrated in the Fig. 9. F_O represents the influence of the obstacle on the robot and F_{BC} is the force for driving the robot. This means that if the control algorithm is based on the artificial potential filed [18], then only F_O is driving the robot. As a result, the robot can no longer move away from this location and thus gets trapped. By contrast, if the proposed BC algorithm is used for control the swarm robots, $F_{BC} > 0$, and the robot will be able to move out of the local minimum and continue searching the area. The above example demonstrates that the BC algorithm has a better capability of escaping from local optimums than a potential field based algorithm.

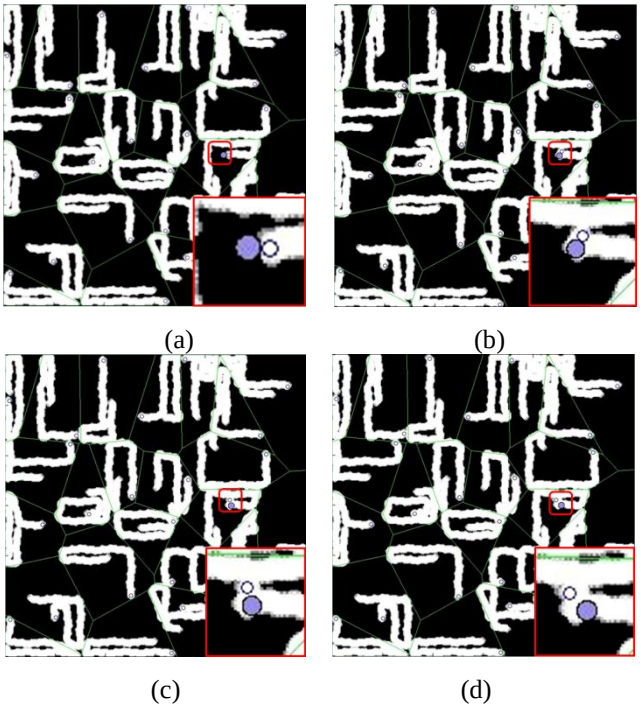


Fig. 8 Collision avoidance with a moving obstacle by 26 robots. The purple circle represents the moving obstacle and the smaller one represents a robot.

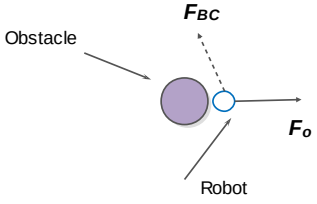


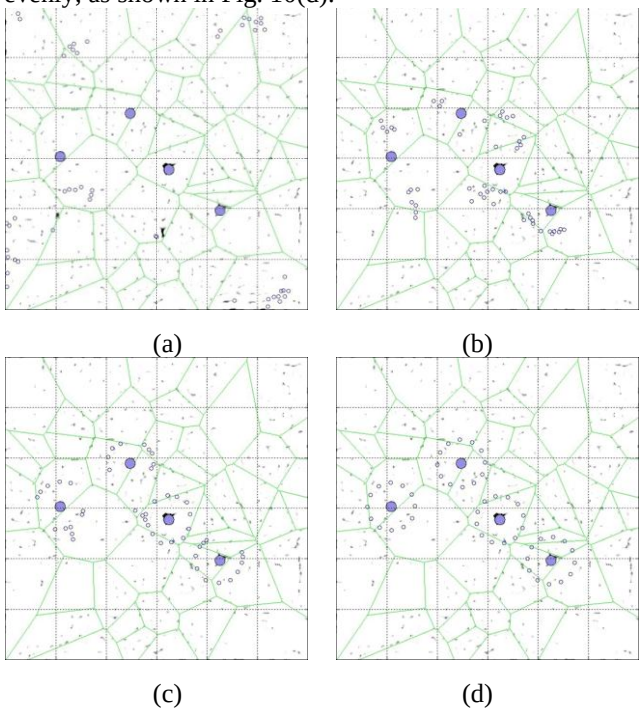
Fig. 9 The force diagram for the trapped robot.

4.2. Multi-Target Trapping

Multi-target trapping by swarm robots is investigated here to further assess the performance of the proposed algorithm. We consider a scenario where 48 robots are used to trap four targets using the BC algorithm. In this

simulation, robots detect the targets and trap each target using the same number of robots.

The results for multi-target trapping by swarm robots are illustrated in Fig. 10. We can find that each target is trapped by 12 robots and the distance between the neighbouring robots can be tuned automatically by using Algorithm 2. When the robots turn into leading state, as shown in Fig. 10 (a), they separate into different groups for detecting the moving targets, as shown in Fig. 10(b). The robots will then switch into the trapping state once they detect targets, which are illustrated in Fig. 10 (c). Finally, the positions of the robots are tuned so that they distribute evenly, as shown in Fig. 10(d).



(a) (b)

(c) (d)

Fig. 10 Four targets trapping using 48 robots**.**

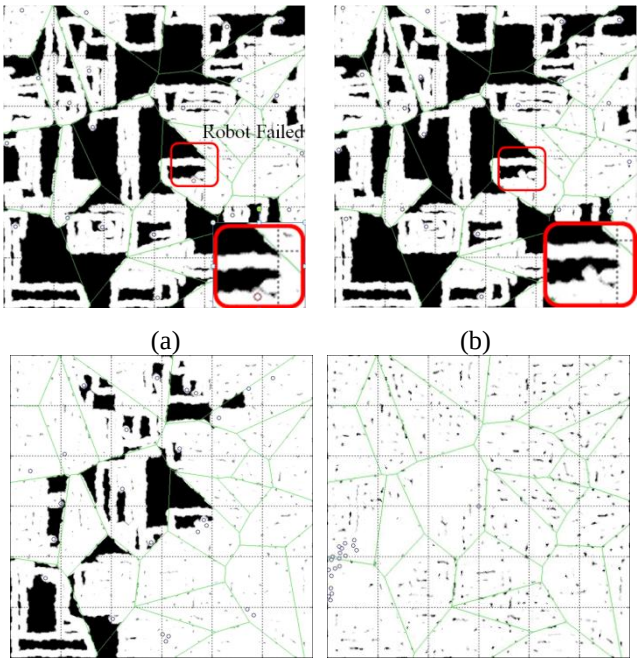
4.3. Target Searching and Trapping in the Presence of Robot Failure

Finally, we evaluate the robustness of the system if some robots become defect during accomplishing the task. For this purpose, we assume one robot is defect during the search (the one marked with red rectangle in Fig. 11).

From Fig.11 (a), we can see that all robots work properly for target searching. Then, we assume the robot marked by a red rectangle breaks down. The consequent reactions of the swarm robotic system controlled by the BC algorithm are shown in Fig. 11 (b)-(d). We can see that other robots are able to autonomously take over the area originally covered by the defected robot, because the concentration in this area will remain high. Moreover, the robots are able to adjust their position so that an even distribution of the robots can be maintained.

4.4. Interaction Analysis to the Robots

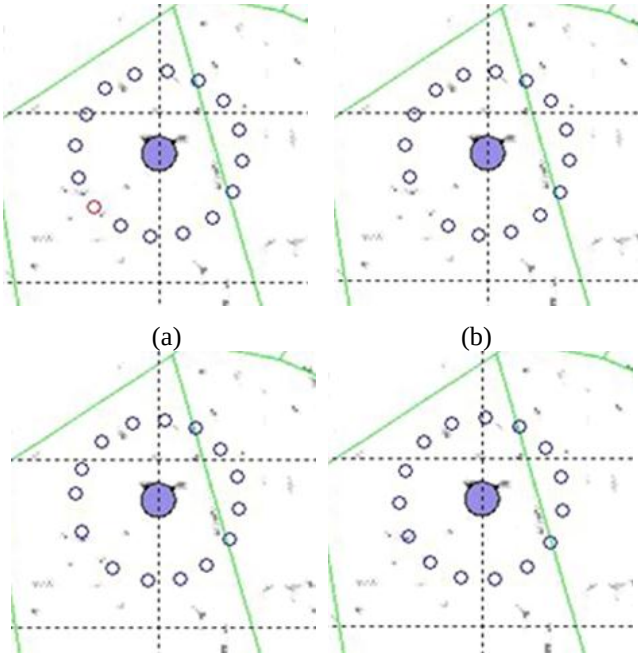
From Fig. 10 and Fig. 12, we find that robots can trap the targets driven by the BC algorithm. However, the distance between the target and the robots, the radius of the



(a) (b)

(c) (d)

Fig.11. Area coverage after a robot's failure. The robot in the red rounded rectangle is defect and the other robots continue to cover the area.



(a) (b)

(c) (d)

Fig.12. Target trapping after a robot's failure by 16 robots.

circle formed by the robots is larger than we set. The reason is that each robot is influenced by its neighbors as well as the target. As the total number of robots in the system changes, the distance between the robots will also be adjusted autonomously. From Fig. 12, we can see that the robot marked in red in Fig. 12(a) was missing in Fig. 12(b) after the trapping mission was completed. Then the other robots will move autonomously under the control of the BC algorithm to locate evenly, as evident from Fig. 12(c) to Fig. 12(d).

In Fig. 13, the red circles represent the robots marked by R_1, R_2 and R_3, and the blue circle stands for the target marked by T_0. R_r is the radius of circle formed by the robots, R_s initialized radius. F_1, F_3 and F_{TR} are forces of R_2 from its neighbouring robots R_1, R_3 and the target T_0. The desired distance from the target to the robots is R_s and the real distance is R_r. Here,

$$F_{TR} \propto R_S \tag{8}$$

To hold the stability of the trapping robots, we can get:

$$F_{TR} = F_{13} \tag{9}$$

where F_{13} is the resultant force of F_1 and F_3. When we ignore the interactions between the robots:

$$\begin{aligned} &\| F_1\| + \| F_3\| = 0 \\ &\Rightarrow F_{TR} = F_{13} = 0 \end{aligned} \tag{10}$$

robot R_2 will trap the target at the distance of R_s without the forces F_1 and F_3. However, the robots interact with their neighbors to avoid collisions following Eq. (2) and (3). So we have

$$\begin{aligned} &\| F_1\| + \| F_3\| > 0 \\ &\Rightarrow F_{13} > 0 \\ &\Rightarrow F_{TR} > 0 \\ &\Rightarrow Rr > Rs \end{aligned} \tag{11}$$

Eq. (11) explains why the radius R_r of the formed circle is longer than the set radius R_s.

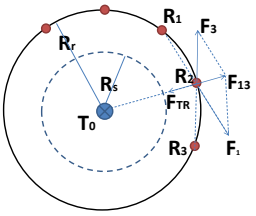


Fig.13. Robots stress analysis when convergence.

4.5. Comparison with other Algorithms

We compare the proposed BC algorithm with some other ones, such as the SACOdm [19], the MSAC[20][21], and the RDPSO [22] for target search. The comparative results are presented in Fig.14. In the comparison, the field size is set to $M \times M$, M=10,20,…,50, and the number of obstacles is 2M.

The SACOdm algorithm exhibits a good performance in search for small areas with, however, however, an increased time as the size of the areas increases.. The comparison, 50 ants are used, and evaporation is set to 0.3, $\alpha = 1, \beta = 7$. The SACOdm algorithm always searches for the shortest path in the area so the robot moves very fast when the field size is small (M=10 or M=20).

By contrast, the time consumed by RDPSO, MSAC and the BC algorithm remains similar when the area size changes. These algorithms are decentralized methods, so the simulation time corresponds to the local information around the robots and the path to the target is formed by the local optimal solutions. In addition, RDPSO needs more time to complete the mission than MSAC and the BC algorithm. RDPSO does not assume global communication

TABLE II
SUMMARY OF SWARM ALGORITHMS FOR SEARCH TASKS

	SACOdm [19]	MSAC [20][21]	RDPSO [22][23][24]	The Proposed BC
Computational complexity	$O(Nt)$	$O(2Nt)$	$O(2Ns)$	$O(Nt)$
Memory complexity	$O(r_\alpha)$	$O(Nt)$	$O(r_\alpha)$	$O(Nt)$
Initial environment	Known	Unknown	Unknown	Unknown
Initial deployment	Fixed	Random	EST approach	Random
Avoid local minima	Low-level control	Local penalty functions	Punish-reward method based on natural selection	Local penalty functions
Obstacle avoidance	Low-level control	Low-level control	Artificial repulsion	Artificial repulsion

PS: r_α stands for truncation of the fractional order series [24].

and as a result, the robots move in a random direction to search targets..

From Fig. 14, we can also find that the time consumption of MSAC and the BC algorithm is almost the same. However, the MSAC algorithm needs the target's information for bidirectional search and the computational complexity is higher, as shown in Table II.

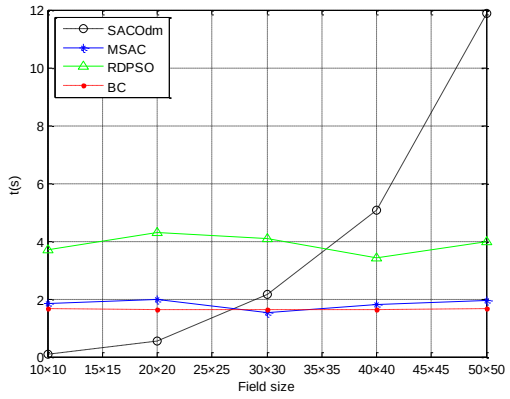


Fig.14. Comparison of robot swarms controlled by four bio-inspired algorithms.

Then we compared the BC algorithm with the GRN algorithm[34] to illustrate the efficiency of trapping one target. The field size is set to 30×30 and the initial positions for the robots and target are randomly initialized in the field. We performed 30 independent runs for each simulation for the BC and GRN algorithms, respectively. The results are shown in Fig. 15. We can find that the time cost for the BC algorithm is less than that of the GRN algorithm. The reason is that the GRN based algorithm controls the robots for two steps mediated by two protein concentrations while the BC algorithm uses just the gradient to guide the movements. The time consumption in different runs changes a lot for both the BC and GRN algorithms depending on the randomly initialized positions of the swarm robots and the target.

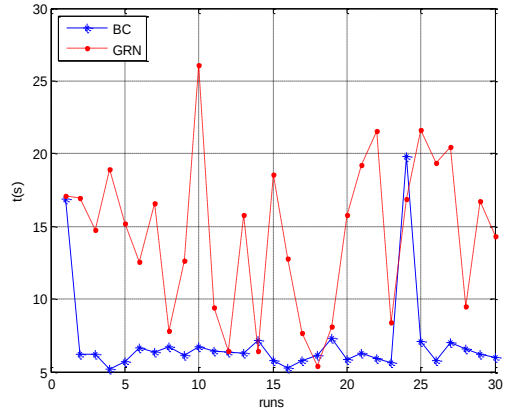


Fig.15. Trapping comparison for the BC and GRN algorithms.

While the proposed algorithm can perform both trapping and search, traditional algorithms like virtual structure and leader-following algorithms can accomplish trapping tasks only. In addition, the BC algorithm is more flexibile than the leader-following algorithm because in the BC algorithm, each robot does not need to follow a fixed leader or relative leader. Due to the above differences, direction performance comparison is not likely..

5. Conclusions and Future Work

In this paper, we present a distributed control algorithm of swarm robot for target search and trapping inspired by bacteria chemotaxis. Extensive simulations have been performed to assess the performance, robustness and computational complexity of the algorithm. Our results have demonstrated the effectiveness and robustness of the BC algorithm compared with several state-of-the-art methods for distributed control of swarm robots.

For the future work, we will further study the robustness of the proposed algorithm in the presence of various disturbances in swarm robots. In addition, we will examine performance change if some of the assumptions, for example, the global communication can only be partially satisfied. Finally, we will implement the BC algorithm on physical robots to verify the performance.

ACKNOWLEDGMENTS

This work was supported in part by the Key Project of the National Nature Science Foundation of China (No. 61134009), the National Nature Science Foundation of China (No. 61473078), Cooperative research funds of the National Natural Science Funds Overseas and Hong Kong and Macao scholars (No. 61428302), Program for Changjiang Scholars from the Ministry of Education, Specialized Research Fund for Shanghai Leading Talents, Project of the Shanghai Committee of Science and Technology (Nos. 13JC1407500), and Innovation Program of Shanghai Municipal Education Commission (No. 14ZZ067).

References

[1] V. Trianni and S. Nolfi, "Self-organizing sync in a robotic swarm: a dynamical system view," *Ieee Transactions on Evolutionary Computation*, vol. 13, pp. 722-741, Aug 2009.

[2] B. Kuipers and B. YungTai, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Robotics and Autonomous Systems*, vol. 8, pp. 47-63, Nov. 1991.

[3] Y. Gabriely and E. Rimon, "Spanning-tree based coverage of continuous areas by a mobile robot," *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 77-98, 2001.

[4] P. Dasgupta, T. Whipple, and C. Ke, "Effects of multi-robot team formations on distributed area

coverage," *International Journal of Swarm Intelligence Research,* vol. 2, pp. 44-69, Jan.-March 2011.

[5] Y. Stergiopoulos and A. Tzes, "Spatially distributed area coverage optimisation in mobile robotic networks with arbitrary convex anisotropic patterns," *Automatica,* vol. 49, pp. 232-237, Jan 2013.

[6] F. Xiao, L. Wang, J. Chen, and Y. P. Gao, "Finite-time formation control for multi-agent systems," *Automatica,* vol. 45, pp. 2605-2611, Nov 2009.

[7] Y. Meng, H. Guo, and Y. Jin, "A morphogenetic approach to flexible and robust shape formation for swarm robotic systems," *Robotics and Autonomous Systems,* 2012.

[8] N. Ghods and M. Krstic, "Multiagent deployment over a source," *IEEE Transactions on Control Systems Technology,* vol. 20, pp. 277-285, Jan 2012.

[9] Z. J. Ji, Z. D. Wang, H. Lin, and Z. Wang, "Interconnection topologies for multi-agent coordination under leader-follower framework," *Automatica,* vol. 45, pp. 2857-2863, Dec 2009.

[10] H. Rezaee and F. Abdollahi, "A decentralized cooperative control scheme with obstacle avoidance for a team of mobile robots," *IEEE Transactions on Industrial Electronics,* vol. 61, pp. 347-354, Jan 2014.

[11] M. A. Lewis and K. H. Tan, "High precision formation control of mobile robots using virtual structures," *Autonomous Robots,* vol. 4, pp. 387-403, 1997.

[12] W. Ni and D. Z. Cheng, "Leader-following consensus of multi-agent systems under fixed and switching topologies," *Systems & Control Letters,* vol. 59, pp. 209-217, Mar-Apr 2010.

[13] A. H. Hu, J. D. Cao, M. F. Hu, and L. X. Guo, "Consensus of a leader-following multi-agent system with negative weights and noises," *IET Control Theory and Applications,* vol. 8, pp. 114-119, Jan 2014.

[14] J. H. Qin, C. B. Yu, and H. J. Gao, "Coordination for linear multiagent systems with dynamic interaction topology in the leader-following framework," *IEEE Transactions on Industrial Electronics,* vol. 61, pp. 2412-2422, May 2014.

[15] W. Zhu and D. Z. Cheng, "Leader-following consensus of second-order agents with multiple time-varying delays," *Automatica,* vol. 46, pp. 1994-1999, Dec 2010.

[16] X. Y. Luo, N. N. Han, and X. P. Guan, "Leader-following formation control of multi-agent networks based on distributed observers," *Chinese Physics B,* vol. 19, Oct 2010.

[17] O. Cetin, I. Zagli, and G. Yilmaz, "Establishing obstacle and collision free communication relay for UAVs with artificial potential fields," *Journal of Intelligent & Robotic Systems,* vol. 69, pp. 361-372, Jan 2013.

[18] W.-r. Shi, X.-h. Huang, and W. Zhou, "Path planning of mobile robot based on improved artificial potential field," *Journal of Computer Applications,* vol. 30, pp. 2021-3, Aug. 2010.

[19] M. A. P. Garcia, O. Montiel, O. Castillo, R. Sepulveda, and P. Melin, "Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation," *Applied Soft Computing,* vol. 9, pp. 1102-1110, Jun 2009.

[20] Q. B. Zhu, J. Hu, W. B. Cai, and L. Henschen, "A new robot navigation algorithm for dynamic unknown environments based on dynamic path re-computation and an improved scout ant algorithm," *Applied Soft Computing,* vol. 11, pp. 4667-4676, Dec 2011.

[21] Q. Zhu, J. Hu, and L. Henschen, "A new moving target interception algorithm for mobile robots based on sub-goal forecasting and an improved scout ant algorithm," *Applied Soft Computing,* vol. 13, pp. 539-549, Jan 2013.

[22] M. S. Couceiro, J. A. T. Machado, R. P. Rocha, and N. M. F. Ferreira, "A fuzzified systematic adjustment of the robotic Darwinian PSO," *Robotics and Autonomous Systems,* vol. 60, pp. 1625-1639, Dec 2012.

[23] M. S. Couceiro, C. M. Figueiredo, R. P. Rocha, and N. M. F. Ferreira, "Darwinian swarm exploration under communication constraints: Initial deployment and fault-tolerance assessment," *Robotics and Autonomous Systems,* vol. 62, pp. 528-544, Apr 2014.

[24] M. S. Couceiro, P. A. Vargas, R. P. Rocha, and N. M. F. Ferreira, "Benchmark of swarm robotics distributed techniques in a search task," *Robotics and Autonomous Systems,* vol. 62, pp. 200-213, Feb 2014.

[25] K. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine,* vol. 22, pp. 52-67, 2002.

[26] R. A. Hooshmand and S. Soltani, "Fuzzy optimal phase balancing of radial and meshed distribution networks using BF-PSO algorithm," *IEEE Transactions on Power Systems,* vol. 27, pp. 47-57, 2012.

[27] N. A. Okaeme and P. Zanchetta, "Hybrid bacterial foraging optimization strategy for automated experimental control design in electrical drives," *IEEE Transactions on Industrial Informatics,* vol. 9, pp. 668-678, May 2013.

[28] S. Muller, J. Marchetto, S. Airaghi, and P. Kournoutsakos, "Optimization based on bacterial chemotaxis," *IEEE Transactions on Evolutionary Computation,* vol. 6, pp. 16-29, 2002.

[29] H. Guo, Y. Jin, and Y. Meng, "A morphogenetic framework for self-organized multirobot pattern formation and boundary coverage," *ACM Transactions on Autonomous and Adaptive Systems,* vol. 7, pp. 1-23, 2012.

[30] A. Breitenmoser, "Voronoi coverage of non-convex environments with a group of networked robots," *IEEE International Conference on Robotics and*

Automation, Anchorage, AK, United states, May 3 - 7, 2010, pp. 4982-4989.

[31] M. Kloetzer and C. Belta, "Temporal logic planning and control of robotic swarms by hierarchical abstractions," *IEEE Transactions on Robotics*, vol. 23, pp. 320-330, Apr 2007.

[32] J. Rekimoto, T. Miyaki, and T. Ishizawa, "LifeTag: WiFi-based continuous location logging for life pattern analysis," *Location- and Context-Awareness*, vol. 4718, pp. 35-49, 2007.

[33] M. Amir, S. Bedra, S. Benkouda, and T. Fortaki, "Bacterial foraging optimisation and method of moments for modelling and optimisation of microstrip antennas," *Microwaves, Antennas & Propagation, IET*, vol. 8, pp. 295-300, 2014.

[34] Y. Jin, Y. Meng, and H. Guo, "A morphogenetic self-organization algorithm for swarm robotic systems using relative position information," *2010 UK Workshop on Computational Intelligence*, Colchester, United kingdom, Sep. 8-10, 2010.