An adaptive particle swarm optimization algorithm for distributed search and collective cleanup in complex environment

Cai, Yi; Chen, Zhutian; Li, Jun; Li, Qing; Min, Huaqing

Published in:
International Journal of Distributed Sensor Networks

Published: 01/01/2013

Document Version:
Final Published version, also known as Publisher's PDF, Publisher's Final version or Version of Record

License:
CC BY

Publication record in CityU Scholars:
Go to record

Published version (DOI):
10.1155/2013/560579

Publication details:
Cai, Y., Chen, Z., Li, J., Li, Q., & Min, H. (2013). An adaptive particle swarm optimization algorithm for distributed search and collective cleanup in complex environment. *International Journal of Distributed Sensor Networks*, *2013*, Article 560579. https://doi.org/10.1155/2013/560579

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2013, Article ID 560579, 9 pages
http://dx.doi.org/10.1155/2013/560579

Research Article

An Adaptive Particle Swarm Optimization Algorithm for Distributed Search and Collective Cleanup in Complex Environment

Yi Cai,[1] Zhutian Chen,[1] Jun Li,[2] Qing Li,[3] and Huaqing Min[1]

[1] *School of Software Engineering, South China University of Technology, Guangzhou 510006, China*
[2] *Information Science and Technology School, Zhanjiang Normal University, Zhanjiang 524299, China*
[3] *Department of Computer Science, City University of Hong Kong, Hong Kong*

Correspondence should be addressed to Huaqing Min; huaq.min@gmail.com

Received 13 November 2013; Accepted 20 November 2013

Academic Editor: Fatos Xhafa

Distributed coordination is critical for a multirobot system in collective cleanup task under a dynamic environment. In traditional methods, robots easily drop into premature convergence. In this paper, we propose a Swarm Intelligence based algorithm to reduce the expectation time for searching targets and removing. We modify the traditional PSO algorithm with a random factor to tackle premature convergence problem, and it can achieve a significant improvement in multi-robot system. It performs well even in a obstacle environment. The proposed method has been implemented on self-developed simulator for searching task. The simulation results demonstrate the feasibility, robustness, and scalability of our proposed method compared to previous methods.

1. Introduction

Searching and cleaning targets in an unknown dynamic environment through multirobot system have numerous real world applications [1], such as hazardous waste cleanup [2], urban search and rescue [3], surveillance systems, and monitoring in military combat environment. Distributed multirobot systems demand group coherence and group competence [4]. Due to its robustness, flexibility, and reliability, the distributed coordination is desirable for multirobot systems under dynamic environment [5, 6]. Therefore, the main challenge for multirobot systems is to develop intelligent robots which can adapt their behaviors based on interaction with the environment and other robots, so as to become more proficient in their tasks in new situations [7, 8].

Recently, when tackling the problem of how to organize the large scale robots efficiently without high cost under dynamic and unknown environment, more and more researchers adopt bioinspired coordination methods. Swarm robotics provides a new approach for coordination of multirobot systems consisting of large number of relatively simple robots. The term “Swarm Intelligence” is inspired by an understanding of the decentralized mechanisms that underlie the organization of natural swarms such as ants, bees, fish, wolves, and even humans.

Typical research in swarm-based robotic systems can be classified to a few domains, which include biological inspiration [9], parallel searching [10, 11], coordinated movement [12], pattern formation [13, 14], and mapping and localization [15]. All of these systems consist of multiple robots or embodied simulated robots acting autonomously based on their own decisions.

To stimulate the colonial organization, such as bird flock and ant colony, Swarm Intelligence theory has been put forward. Particle Swarm Optimization and Ant Colony Optimization are two of the most representative algorithms based on the Swarm Intelligence theory. Also, some researches [16] use these two algorithms to organize large scale multirobot system in searching and cleanup tasks. With these methods, a robot in a multirobot system can make its own decision based on its local environment information and interaction with its neighbors without centralized control. However, since this kind of algorithms’ characteristics, using them to organize and handle complex task has the problems of low convergence speed and sensitivity to local convergence. Robots are easily plunged into local minimum and hard to explore new area, and thus the global performance of multirobot systems may be greatly reduced.

In the collective cleanup task, there are some targets randomly distributed in target area. The robots may also be randomly located in the target area. They should find targets first and then remove them. The mass of target is stochastic. When some targets are too big for a single robot to cleanup individually, it needs to cooperate with other robots in the environment. Therefore, a robot needs to judge whether it can handle the task individually or cooperate with other robots, so as to handle the collective cleanup task efficiently and effectively. There might be some obstacle [17] spread in the target map; robot cannot go through the obstacle; they have to avoid the obstacle.

To address this problem, we propose a modified PSO algorithm trying to minimize searching time and handle the premature convergence problem which has not been tackled by most of researchers previously. In our method, the searching area is divided into two-levels; that is, the task map is divided into distinct subareas and each subarea is divided into distinct grids. A collective cleanup task can be divided into two phases, that is, searching phases and removing phase. When there is no detected target in robot's target list, the robot will be in searching phase. If a robot discovers a target or receives message about targets from other robots, it will switch its state from searching to removing. No matter in which phase the robot is, a corresponding utility function is given for deciding the next movement. A Particle Swarm Optimization based algorithm has been proposed to balance the robot's exploration and exploitation behavior through interaction among robots. We also modify the algorithm with a random factor to tackle premature convergence problem, and it can make a significant improvement in global performance of the multirobot system. There is no centralized control unit; each robot just makes decision individually by its local information and communication with its neighbors. We use a Wi-Fi model for robots' communication between each other. To evaluate the proposed method, we also conduct simulations by comparing previous methods.

The paper is organized as follows. Section 2 introduces the related work. Section 3 describes the proposed methodology. We conduct simulation and present the simulation results in Section 4. To conclude the paper, Section 5 outlines the research conclusions and future work.

2. Related Work

Within the field of multirobot systems, multirobot search and cleanup is one area which is currently receiving a great quantity of research attention. In 1987, Reynolds [18] modeled the motion of a flock of birds and used computer to simulate it. Payton et al. [19] propose a method for coordinating the actions of large numbers of small-scale robots through using virtual pheromone named symbolic message, which is modeled after the chemical insects, such as ants. Michael et al. [20] use market-based coordination protocols algorithm to dynamically assign tasks to multirobots. He makes an assumption that every robot has knowledge of the maximum number of robots.

The method inspired by PSO (Particle Swarm Optimization) algorithm [21] has received much attention in the last few years. Particle Swarm Optimization algorithm is designed through stimulation of a simplified model of flock of animals such as birds.

Pugh and Martinoli [22] present a search algorithm inspired by PSO. They exploit this inspiration by modifying the PSO to mimic the multirobot search process. Each robot is represented by a particle in a two-dimensiona search space. However, the algorithm only focuses on single target searching, and it is hard to extend this algorithm into multitarget searching. Hereford et al. [23] propose a distributed PSO method for a multirobot searching task. Each robot updates its own position and velocity based on its local measurements and the global best position shared with others. Besides, this algorithm is only applied to a single target searching.

Doctor et al. [24] study the multitarget searching case by dividing robots into subgroups and each group is dedicated to one target searching. A global processing is conducted to identify the most important target once all of the targets have been identified.

Meng and Gan [25] present a PSO-inspired algorithm to combine explorative searching and dynamic task allocation together for collective construction. They use this algorithm to balance the tradeoff of exploration and exploitation when multirobots were assigned to the collective cleanup task. A grid-based map is used for target searching and each robot has to store the global map's grids which mean the robot needs enough large memory. Later, they improve their works [26] by dividing the robot status into two phases: searching and selecting, where each phase has its own utility function.

Liu et al. [16] propose a modified PSO algorithm that minimizes the time for searching by dividing the global map into two-level subareas so each robot only need, to store local grid information, which can reduce communication traffic when robots interact with each other. They use Wi-Fi for robots' interaction to avoid significant overlap during exploration.

Couceiro et al. [27] proposed two modified versions of the Particle Swarm Optimization and the Darwinian PSO(DPSO) algorithms, based on obstacles avoidance abilities and real-world multirobot systems (MRS). The concepts of social exclusion and social inclusion are used in the robotic DPSO algorithm as a punish-reward mechanism enhancing the ability to escape from local optima.

Masr and Zelenka [28] modified the PSO algorithm and proposed fitness function for such purpose and modified a basic principle which controls movements of the particles. According to their works, the most important feature of their work is the fact that using such simple rules is satisfactory to lead robots to explore their surroundings.

In our previous works [29], we modify the Particle Swarm Optimization method to balance the tradeoff of exploration and exploitation. Now we extend our work to new environment with obstacles. Also we proposed a more effective optimization strategy to handle the premature convergence in multitarget searching problem when using PSO algorithm. Robots communicate with each other through Wi-Fi, and each robot makes its own decision individually by local information and interactions with other robots.

3. Methodology

Initially, each robot and target randomly distributed in the environment, and no targets' or robots' position overlap. It is efficient to search the local subarea first and then others. Basically, we divide the cleanup task into two phases: searching and removing; each phase has its own utility function. The states of robots are all set to searching initially.

During the searching phase, each robot detects its local subarea in the beginning. If a subarea has not been detected yet, the robot is going to decide which subarea to be detected next by itself according to local environment information and interaction with its neighbors. The robot should detect the undetected subarea as much as possible in order to speed up the task. Once a robot discovers a target by itself or receives message about target from its neighbor, it will switch its state from searching to removing, and it has to balance the tradeoff of exploration and exploitation. If there is only one target in the robot's target list, it has to decide whether it should move to remove the target or continue detecting other undetected subarea. If there is more than one target in the robot's target list, it not only needs to decide what it should do but also has to decide which target to be removed once it chooses to remove.

3.1. Map. We suppose that the task area is a grid-based map. The global map is divided into $M \times N$ subareas. We use $S(i, j)$ to represent the state of the subarea whose location is (i, j):

$$S_{ij} = \begin{cases} 1, & \text{subarea } (i, j) \text{ has been detected,} \\ 0, & \text{subarea } (i, j) \text{ has not been detected.} \end{cases} \quad (1)$$

Each subarea consists of equal numbers of grids, which means subarea is divided into grids. We use G_{kt} to record the state of the grid at (k, t):

$$G_{kt} = \begin{cases} 1, & \text{grid } (k, t) \text{ has been detected,} \\ 0, & \text{grid } (k, t) \text{ has not been detected.} \end{cases} \quad (2)$$

Each robot has two buffers. One of them stores the grids value of current subarea so the robot could know which grid has been detected and which has not. When the detected grids of current subarea reach a percentage, the robot will set this subarea to the state that has been detected. Then the robot will pick up another subarea and move to explore it. This buffer then will be used to store new subareas grids state when it moves into a new subarea. Another buffer stores all the subareas values.

3.2. Cooperation

3.2.1. Searching. When a robot is in searching phase, it may face two different conditions: the case that current subarea has not finished detection and the case that detection of current subarea has been finished. Robots adopt corresponding strategy to determine the next movement under different conditions.

In the case that current subarea has not finished detection, a robot should explore its current subarea as fast as possible, meanwhile reducing the overlap of grid detection as much as possible. The simplest method to complete the detected task is letting the robots wander in subarea. But it is obvious that the efficiency of this method is too low to tolerate, and the robots would repeat detecting some grids again and again. In our works, we give each grid a utility function. The higher the utility of a grid, the higher priority a robot will go to detect. According to the observation they [25] did, it will be more effective to let robot detect the corner of a subarea first. Hence we defined the utility function as

$$U_{xy}^{\alpha}(t) = D_{xy}^{ij}, \quad (3)$$

where $U_{xy}^{\alpha}(t)$ means the utility of grid (x, y) for robot α in time t and D_{xy}^{ij} represents the distance between grid (x, y) in subarea (i, j) and the center of subarea (i, j).

Robot will mark the grid of current subarea which has been detected yet and store it in its own buffer, and it would not move to detect this kind of grid. So the overlap problem can never happen. Besides, it can minimize the expected time for searching; therefore global performance will be better. When the percentage of detected grids arrives at a user setting value, the robot will set the state of this subarea to 1 in its buffer, and then it will move to detect other subareas.

In another case, robot completes detecting current subarea and it is going to determine which subarea to be detected in next move. We believe that a robot should follow two factors, in order to accelerate the task, while choosing the next subarea to be detected: (1) the travel cost to the subarea should be low; (2) the potential of having target in the subarea should be high. We make an assumption that all the robots know how many undetected grids a subarea has, which can be represented by $\sum G_{xy}^{ij}$, to reduce the complexity of computation. According to Meng's work [26], the probability of having a target in subarea (i, j) at time t can be defined as

$$P_{ij}(t) = \frac{D_{ij}}{\sum_{m=1, n=1}^{M,N} D_{mn}} \times \sum G_{xy}^{ij}, \quad (4)$$

where D_{ij} is the distance between subarea (i, j) and map center.

Now we can define the utility function of each subarea when robot needs to choose which subarea to be detected next. Based on the two factors we mentioned before, the utility function of subarea (i, j) for robot can be defined as

$$\mu_{ij}^{\alpha} = \frac{P_{ij}(t)}{D_{ij}^{\alpha}(t)}, \quad (5)$$

where $P_{ij}(t)$ is the probability of having a target in subarea (i, j) and $D_{ij}(t)$ means the distance between robot and the center of subarea (i, j). The same as before, the higher the utility of a subarea, the higher the probability a robot will choose it to be detected.

3.2.2. Removing. During removing phase, each robot has its own target list. Once again, we need a utility function of each target in target list for a robot. We suppose that when a robot discovers a target, it can also scan and get the mass of this target. Therefore we can easily define the utility function of a target as

$$\varphi_{ij}^{\alpha} = \frac{W_{ij}(t)}{D_{ij}^{\alpha}(t)}, \quad (6)$$

where $W_{ij}(t)$ is the mass of the target at location (i, j) at time t and $D_{ij}(t)$ represents the distance between robot and this target. In general, the higher the utility of the target is, the more attractive the corresponding target is to the robot. Hence, it will benefit more than other if the robot moves to remove the target with a higher utility.

3.2.3. Communication. We use Wi-Fi model to achieve communication between each robot. Every robot can contact with others when they are in the range of Wi-Fi scope. When two robots are in predefined range, they can interact and change the local information with each other. The traditional methods let the two robots exchange all information. To reduce the cost of interaction, we improve the role of interaction. Each robot only stores the grids' state of current subarea and the states of all subareas in its buffer. When two robots, which are preparing to exchange information, are in the same subarea, they would change the girds' state of current subarea. In addition, they would change the global subareas' state too. If the two robots are in different subareas, they will just exchange the global subareas' state.

3.2.4. Modified PSO. After the discussion above, we have different utility functions under any circumstance. Generally speaking, we can simply use greedy algorithm to choose the position of next move. But the global performance will be so much inefficient because it cannot balance the tradeoff of exploration and exploitation. As we know, the tradeoff needs to be adjusted; it means that the robots should not only be self-interested, but also work together to achieve optimal global performance.

We pay our attention to Particle Swarm Optimization algorithm to handle this problem. According to PSO algorithm, each particle combines the local information obtained by it and global information obtained by its neighbors to update its position and velocity. In our multirobots system, it is hard to compute the global best because each robot just can communicate with the one which is in the predefined range. Inspired by the work of other researchers, we use a modified PSO algorithm to update each robot's position and velocity. The velocity vector and position of a robot can be updated as

$$\begin{aligned} V_i(t+1) &= \omega \times V_i(t) + C_1 \times \text{rand} \times (P_b - X_i(t)), \\ X_i(t+1) &= X_i(t) + V_i(t+1), \end{aligned} \quad (7)$$

where $V_i(t)$ is the velocity of robot i, C_1 is the accelerated factors, ω is the inertia factor, $X_i(t)$ is the position of robot i, and P_b represents the position which has the highest utility computed by corresponding utility function. For instance, if a robot is in removing phase, the P_b = position of $\max \varphi_i j$. As usual, there are upper bound and lower bound of velocity, if the new velocity obtained by the update function is out of the range of predefined scope.

In the application of PSO, the algorithm usually tends to converge prematurely because of the insufficient diversity of individuals. And all the robots are sensitive to local convergence. Once a robot finds a high utility target, it will rush to it and most of the robots will tend to the target because of the interaction between them. Another disadvantage of PSO is that as time goes on the robot has certain possibility to wander in narrow areas and lose the ability to explore.

To deal with this problem, we believe that we can fix it with improving the ability for detecting new areas without considering the utility of each robot. Hence, we propose an approach that works in the process when a robot needs to determine which position it will move in the next step. We modify the function of update velocity as follows:

$$\begin{aligned} V_i&(t+1) \\ &= (1-\rho) \times [\omega \times V_i(t) + C_1 \times \text{rand} \times (P_b - X_i(t))] \\ &\quad + \rho \times [\text{random } V], \end{aligned} \quad (8)$$

where ρ is a $1-0$ two-value number whose value is decided randomly and random V represents a random velocity in the predefined scope.

This formula means that before each robot decides where to move in the next step, it should roll a dice first; if it gets the specific value, it will make a random movement instead of updating velocity under PSO algorithm. Otherwise, it uses PSO to select the next movement as usual.

To enhance the searching ability, we make some change of the inertia factor. According to Shi and Eberhart [30], a big inertia factor is a benefit for global searching and a small inertia factor is a benefit for local searching. We define a time decreasing inertia factor as a function of

$$\omega = t_{\text{start}} - (t_{\text{start}} - t_{\text{end}}) \times \left(\frac{k}{T_{\max}}\right)^2, \quad (9)$$

where k is the current iteration, and $T_{\max}$ is the max iteration, t_{start} and t_{end} are two parameters. Hence, the robot has a good ability to explore new area in a large scale in the initial stage while search local area with high accuracy later.

In reality environment, there might be some obstacles spread in the target area. Robots cannot remove the obstacle and walk through it. If a robot Robot must avoid the obstacle to finish the cleanup task.

In original PSO algorithm based robot, the multirobot system does not handle the obstacle and moves as usual. In our algorithm, we just make a simple modification to the moving strategy of robot, so it can adapt to complicated environment.

(i) If there is an obstacle standing in the way of a robot, the robot gives up the target and finds another target based on the modified PSO algorithm.

If any robot falls into the obstacles area, obstacle avoidance mechanism is activated to find the new target position for the robot. With this adaptive moving strategy, the robot swarm can finish its task as usual even there are obstacles.

3.2.5. Computational Complexity Issues. The computational complexity analysis is concerned with the complexity of the problem over all possible instances, sizes, and algorithms. There has been an attempt to evaluate Particle Swarm Optimization lgorithms. Different from the normal problem solved by PSO algorithm, each robot just makes a little calculation in our method. To obtain the location of next movement, a robot needs to scan its target list and find out the best target with utility function. We suppose that a robot R receives signal of target from N robots. And there are already M targets in the target list of robot R, because the calculation of each target's utility is simple and no loop is involved. Hence, the time cost is $O(N+M)$. This can be verified by simulation experiments.

4. Simulation

4.1. Simulation Set-Up. To evaluate the performance of our modified PSO algorithm in a distributed swarm robot system, we use our self-developed JAVA-based virtual simulator. The simulation environment is as presented below.

(i) The searching environment is a two-dimensional area with 4×4 subareas. Each subarea consists of 10×10 girds, so there are totally 1600 grids in the global map.

(ii) One target or robot just occupied one grid and none of them are overlapped in the same grid. In other words, the grid is the smallest unit in our simulation.

(iii) The local communication radius, which is the Wi-Fi range, of each robot is predefined. And the default value of target visibility range is 3 girds.

(iv) Each robot can only move vertically or horizontally. It is reasonable. If the grid is small enough and the robots occupy multi grids, the robot will move normally. Hence, in order to simplify the simulation, we suppose that each robot can move limitedly.

(v) Population size is set to 10, 15, 20, 25, and 30 depending on the searching environment. In general, evolutionary search algorithm performs better with relatively large population. However, a large number of robots will cost a lot without producing significant improvements.

(vi) We make an assumption that there are totally 400 unit mass uniform distributed across 10 targets. It means that the mass of each target is likely the same.

(vii) Cognitive coefficient C_1 is set to 2 for all robot based on the setting in the experiments in previous studies.

(viii) The coefficients of inertial factor function t_{start} and t_{end} are set to 0.9 and 0.4, respectively, to improve searching performance in different stages.

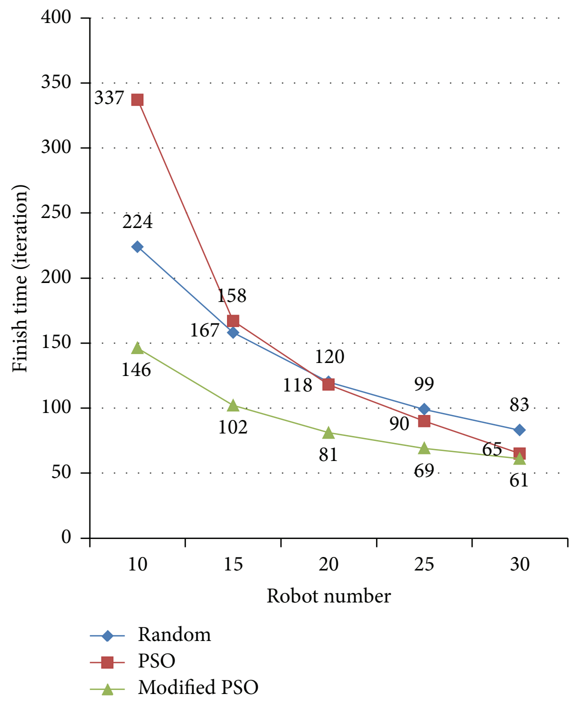

Figure 1: Compared with different methods with different numbers of robots. Wi-Fi range is 4 and detection range is 3.

We repeat simulation over one thousand times on each different condition, and calculate the average iteration and average travel distance when all the targets have been cleaned. At last we choose the traditional PSO-based method and Random-Walk method to compare them with our method.

4.2. Results. Two different situations were simulated in our experiment. The first experiment is the simulation for the task cleanup in a map without obstacle. The second situation is an environment with randomly spread obstacle. Each obstacle has different size.

4.2.1. No Obstacle. Firstly, we assume that there is no obstacle in the map.

In a cleanup task, the population size of robot has a great impact on the performance. No matter which method we use, the number of iterations would be less and less with the population size becoming larger and larger. Hence, it is significant for us to find out a method which can complete the task with robots as little as possible.

According to Figure 1 we can easily find out that, along with the number of robots that increased from 10 to 30, global finish time of all the three method is decreased accordingly. The modified PSO methods has the best performance no matter how many robots are in the multirobot system. Traditional PSO-based method performs as good as modified PSO method when number of robot is large. But on other side, the performance of traditional PSO-based method goes near to the Random-Wander method.

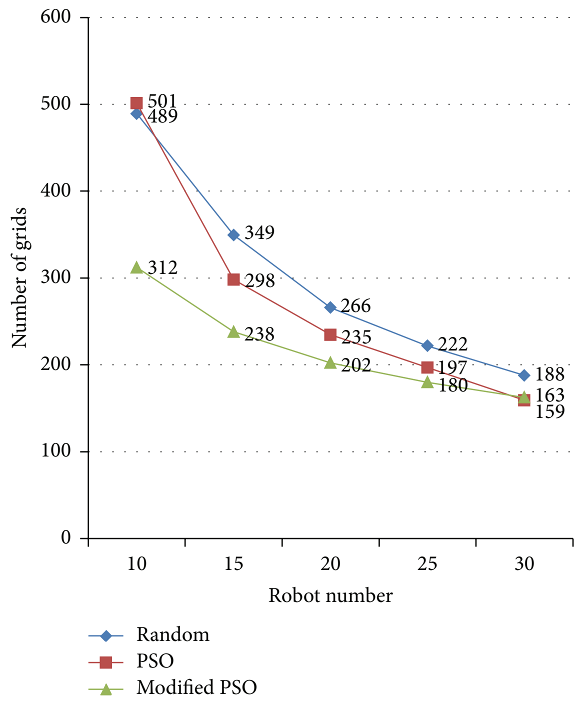

Figure 2: Average travel distance of different method. Wi-Fi range is 4 and detection range is 3.

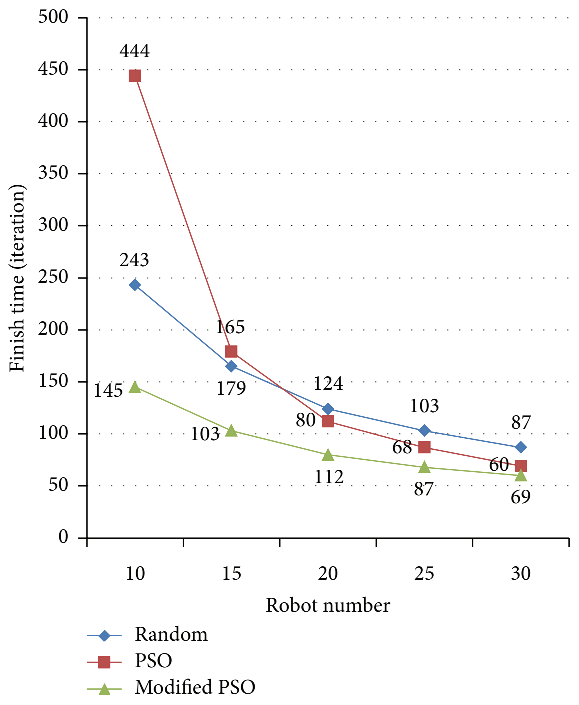

Figure 4: The total iteration of different methods. There are 10 obstacles in the map.

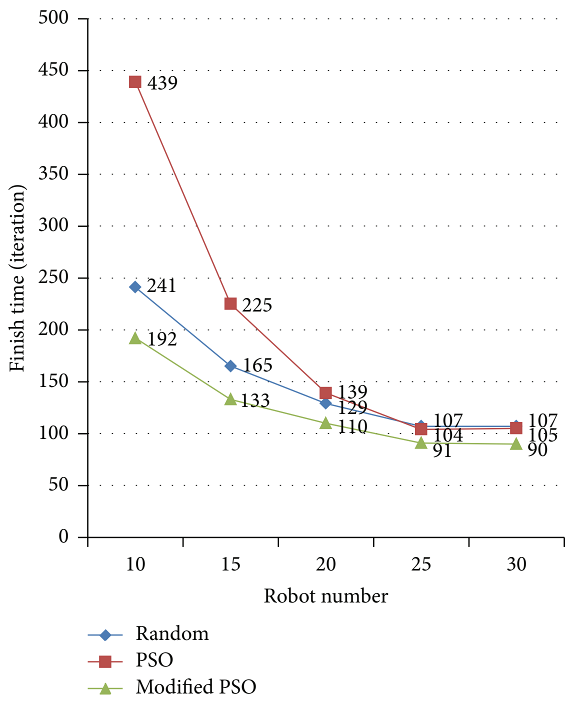

Figure 3: Compared with different methods with different numbers of robots. Wi-Fi range is 4 and detection range is 2.

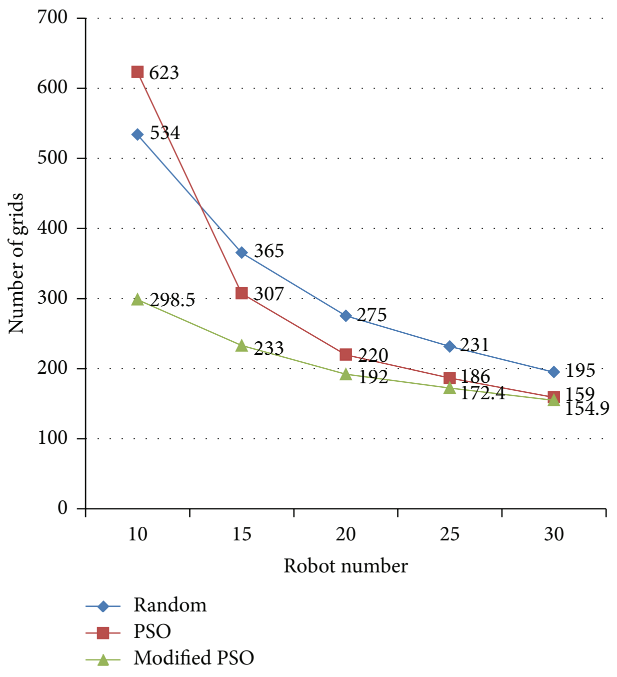

Figure 5: Average travel distance of different methods. There are 10 obstacles in the map.

Figure 2 shows the average travel distance of each robot. As we see, Random-Wander based robot travels the longest distance. PSO-based robot walks less and less as the population size increases. Robot based on our modified PSO always has the best performance. The travel distance is directly related to the overall power consumption of the systems.

From Figure 3 we know that when the target visible range is small, our methods work excellently no matter what size the

Table 1: Task under different conditions.

Wi-Fi range, detection range	PSO	Modified PSO	Random
(2, 2)			
Average	161	110	130
Max	1596	321	348
Min	45	46	60
(4, 2)			
Average	139	110	129
Max	934	396	441
Min	43	46	51
(2, 4)			
Average	68	65	116
Max	283	189	596
Min	35	39	51
(3, 4)			
Average	68	66	114
Max	495	203	332
Min	34	35	51

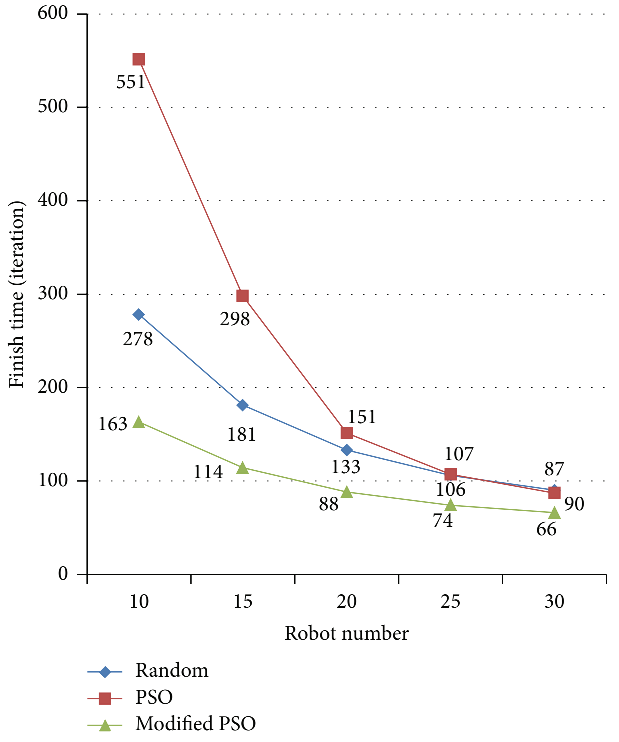

Figure 6: The total iteration of different methods. There are 20 obstacle in the map.

swarm is. But searching by PSO-based method easily drops into premature convergence especially under the small size swarm circumstance. The reason behind this observation is that the robots using PSO are almost greedy and would always try to achieve the best utility. Therefore, they would move to remove targets with highest utilities rather than explore new areas.

Table 1 describes the average and max and min iteration when task has been finished with 20 robots under different circumstances. It is obvious that our modified PSO methods always perform better than other two methods. The min iteration of our method is small enough, while the max iteration is not too big to satisfy us. With the Wi-Fi range and detection range increase, the finish time is limited to a lower bound which is decided by swarm size and target number. On the other side, PSO-based method has a bad performance when the Wi-Fi range and detection range are small. It is reasonable that when these two parameters are small, the robots can only get extremely little information from local environment and neighbors; therefore they tend to search for a narrow area again and again, which means they lost the ability to explore new subarea. Once again, the simulation result illustrates that our modified PSO method outperforms the PSO approach on any occasion.

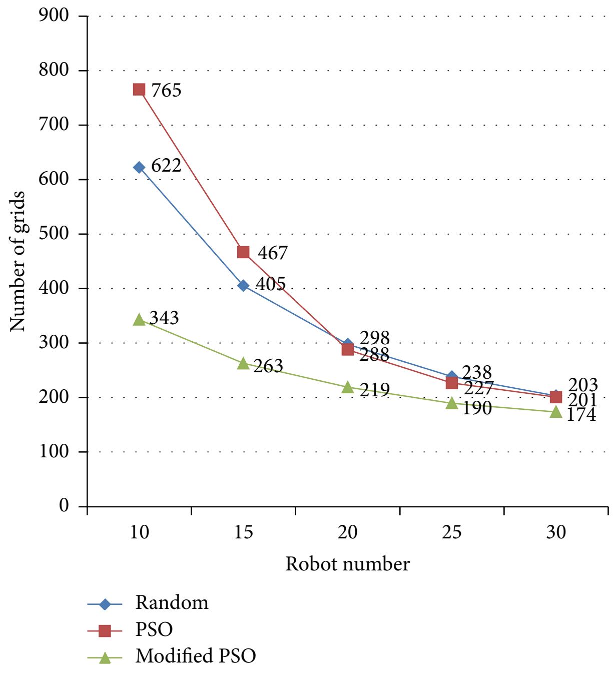

Figure 7: Average travel distance of different methods. There are 20 obstacles in the map.

4.2.2. Environment with Obstacle. In this simulation, some obstacle occupied one or two girds will be spread randomly in the map. We repeat the experiments in the previous section with 10 and 20 obstacles, respectively. As before, the visible range is 4 and the Wi-Fi range is 3.

According to Figures 4, 5, 6, and 7, the robot swarm based on our modified PSO algorithm performs as good as usual. No matter how many the obstacles are, robots can complete the task perfectly. However, the original PSO algorithm based robot has an unsatisfying performance when the number of obstacles is large. It is proved that our algorithm has significant improvement and the robot can adapt to complex

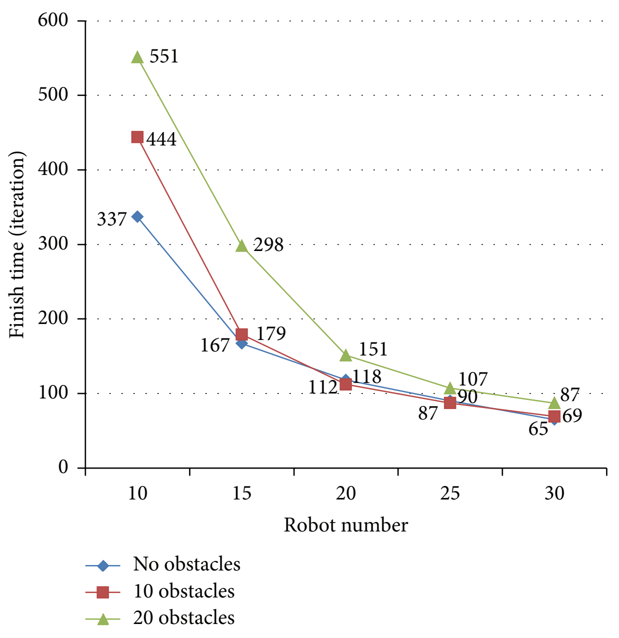

Figure 8: PSO-based robot in different condition.

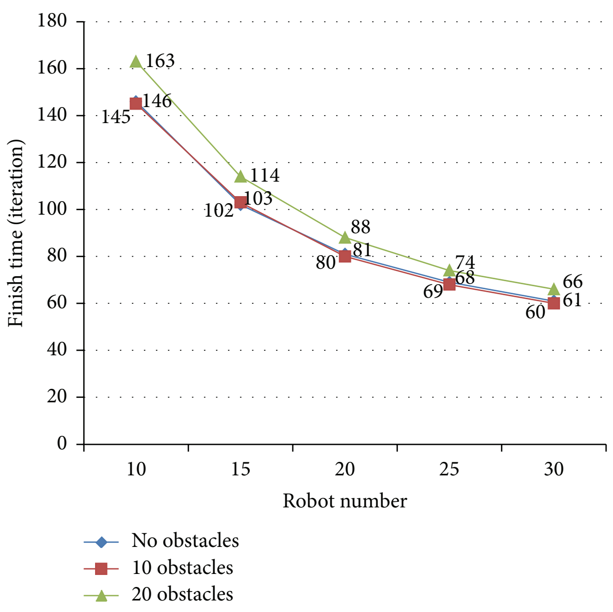

Figure 9: Modified PSO-based robot in different condition.

environment to solve the problem and finish task. The original PSO algorithm has a better performance along with the population size increase. But the performance is much worse than before. In more complicated environment, original PSO algorithm is not suitable.

Figures 8 and 9 show that robot based on PSO performs worst while number of obstacles becomes greater. However, the number of iterations for finishing task is nearly the same. The modified PSO algorithm is flexible enough for different environments.

5. Conclusion

A novel Swarm Intelligence based algorithm is proposed for a distributed search and collective cleanup task. We aim to accelerate multirobots searching and cleaning with a Swarm Intelligence algorithm. Meanwhile, we modified the bioinspired algorithm to handle the problems of low convergence speed and sensitivity to local convergence. Compared with traditional PSO-based algorithm, simulations of the method that we propose present a better performance. The result demonstrates the significant improvement of the new method. One specific embodiment is that the total time of finishing task is more stable and shorter than that of the traditional PSO-based approach. What is more, the average travel distance of each robot is smaller. It means that the overall power consumption of the system is less. Once again, the simulation result illustrates that our modified PSO method outperforms the PSO approach on any occasion.

In the future work, we will consider more about real-world environmental elements and extend our methods to other task for multirobot system. With a new platform, we can experiment our methods in a better angle.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant no. 61300137); the Guangdong Natural Science Foundation, China (no. S2013010013836); the Fundamental Research Funds for the Central Universities, SCUT (no. 2014ZZ0035).

References

[1] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, 1998.

[2] M. J. Mataric, M. Nilsson, and K. T. Simsarian, "Cooperative multi-robot box-pushing," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems "Human Robot Interaction and Cooperative Robots" (IROS '95)*, vol. 3, pp. 556–561, Pittsburgh, Pa, USA, August 1995.

[3] T. Weigel, J.-S. Gutmann, M. Dietl, A. Kleiner, and B. Nebel, "CS freiburg: coordinating robots for successful soccer playing," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 685–699, 2002.

[4] B. Yamauchi, "Decentralized coordination for multirobot exploration," *Robotics and Autonomous Systems*, vol. 29, no. 2-3, pp. 111–118, 1999.

[5] C. A. C. Parker and H. Zhang, "Collective robotic site preparation," *Adaptive Behavior*, vol. 14, no. 1, pp. 5–19, 2006.

[6] M. J. B. Krieger, J.-B. Billeter, and L. Keller, "Ant-like task allocation and recruitment in cooperative robots," *Nature*, vol. 406, no. 6799, pp. 992–995, 2000.

[7] A. Martinoli, A. J. Ijspeert, and F. Mondada, "Understanding collective aggregation mechanisms: from probabilistic modelling to experiments with real robots," *Robotics and Autonomous Systems*, vol. 29, no. 1, pp. 51–63, 1999.

[8] J. Li and K. Kim, "Hidden attribute-based signatures without anonymity revocation," *Information Sciences*, vol. 180, no. 9, pp. 1681–1689, 2010.

[9] T. H. Labella, M. Dorigo, and J.-L. Deneubourg, "Division of labor in a group of robots inspired by ants' foraging behavior," *ACM Transactions on Autonomous and Adaptive Systems*, vol. 1, no. 1, pp. 4–25, 2006.

[10] J. M. Hereford and M. A. Siebold, "Bio-inspired search strategies for robot swarms," in *Swarm Robotics from Biology to Robotics*, E. M. Martinez, Ed., pp. 1–26, InTech, Rijeka, Croatia, 2010.

[11] J. Li, Q. Wang, C. Wang, N. Cao, K. Ren, and W. Lou, "Fuzzy keyword search over encrypted data in cloud computing," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM '10)*, pp. 1–5, San Diego, Calif, USA, March 2010.

[12] A. T. Hayes and P. Dormiani-Tabatabaei, "Self-organized flocking with agent failure: off-line optimization and demonstration with real robots," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '02)*, vol. 4, pp. 3900–3905, May 2002.

[13] C. Zhifu and C. Tianguang, "Aggregation and pattern formation of multi-agent systems," in *Proceedings of the 26th Chinese Control Conference (CCC '07)*, pp. 606–610, Hunan, China, July 2007.

[14] J. Li, Q. Wang, C. Wang, and K. Ren, "Enhancing attribute-based encryption with attribute hierarchy," *Mobile Networks and Applications*, vol. 16, no. 5, pp. 553–561, 2011.

[15] S. Thrun, "Robotic mapping: a survey," in *Exploring Artificial Intelligence in the New Millennium*, The Morgan Kaufmann Series in Artificial Intelligence, chapter 1, pp. 1–35, Morgan Kaufmann, Mateo, Calif, USA, 2003.

[16] D. Liu, X. Zhou, A. Liang, and H. Guan, "A swarm intelligence based algorithm for distribute search and collective cleanup," in *Proceedings of the IEEE International Conference on Intelligent Computing and Intelligent Systems (ICIS '10)*, vol. 2, pp. 161–165, Xiamen, China, October 2010.

[17] A. Z. Mohamed, S. H. Lee, H. Y. Hsu, and N. Nath, "A faster path planner using accelerated particle swarm optimization," *Artificial Life and Robotics*, vol. 17, no. 2, pp. 233–240, 2012.

[18] C. W. Reynolds, "Flocks, herds and schools: a distributed behavioral model," in *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*, vol. 21, no. 4, pp. 25–34, ACM, 1987.

[19] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone robotics," *Autonomous Robots*, vol. 11, no. 3, pp. 319–324, 2001.

[20] N. Michael, M. M. Zavlanos, V. Kumar, and G. J. Pappas, "Distributed multi-robot task assignment and formation control," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '08)*, pp. 128–133, Pasadena, Calif, USA, May 2008.

[21] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, Perth, Wash, USA, December 1995.

[22] J. Pugh and A. Martinoli, "Inspiring and modeling multi-robot search with particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 332–339, Honolulu, Hawaii, USA, April 2007.

[23] J. M. Hereford, M. Siebold, and S. Nichols, "Using the particle swarm optimization algorithm for robotic search applications," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 53–59, Honolulu, Hawaii, USA, April 2007.

[24] S. Doctor, G. K. Venayagamoorthy, and V. G. Gudise, "Optimal PSO for collective robotic search applications," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, vol. 2, pp. 1390–1395, Rolla, Mo, USA, June 2004.

[25] Y. Meng and J. Gan, "LIVS: Local Interaction via Virtual Stigmergy coordination in distributed search and collective cleanup," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '07)*, pp. 1371–1376, San Diego, Calif, USA, October 2007.

[26] Y. Meng and J. Gan, "A distributed swarm intelligence based algorithm for a cooperative multi-robot construction task," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '08)*, pp. 1–6, St. Louis, Mo, USA, September 2008.

[27] M. S. Couceiro, R. P. Rocha, and N. M. F. Ferreira, "A novel multi-robot exploration approach based on Particle Swarm Optimization algorithms," in *Proceedings of the 9th IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR '11)*, pp. 327–332, Kyoto, Japan, November 2011.

[28] M. Masar and J. Zelenka, "Modification of PSO algorithm for the purpose of space exploration," in *Proceedings of the IEEE 16th International Conference on Intelligent Engineering Systems (INES '12)*, pp. 51–54, Lisbon, Portugal, June 2012.

[29] J. Li, Z. Chen, Y. Cai, H. Min, and Q. Li, "A modified Particle Swarm Optimization algorithm for distributed search and collective cleanup," in *Proceedings of the 5th IEEE International Conference on Awareness Science and Technology (iCAST '13)*, pp. 1–6, Aizu-Wakamatsu, Japan, November 2013.

[30] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, Anchorage, Alaska, USA, May 1998.