Hindawi
Complexity
Volume 2020, Article ID 8863526, 12 pages
https://doi.org/10.1155/2020/8863526

WILEY | Hindawi

Research Article

Exploration Enhanced RPSO for Collaborative Multitarget Searching of Robotic Swarms

Jian Yang, Ruilin Xiong, Xinhao Xiang, and Yuhui Shi

Department Computer Science and Engineering, Southern University of Science and Technology (SUSTech), Shenzhen 518055, China

Correspondence should be addressed to Yuhui Shi; shiyh@sustech.edu.cn

Received 7 August 2020; Revised 15 October 2020; Accepted 13 November 2020; Published 28 November 2020

Academic Editor: Zhile Yang

Particle Swarm Optimization (PSO) is an excellent population-based optimization algorithm. Meanwhile, because of its inspiration source and the velocity update feature, it is also widely used in the collaborative searching tasks for swarm robotics. One of the PSO-based models for robotic swarm searching tasks is Robotic PSO (RPSO). It adds additional items for obstacle avoidance into standard PSO and has been applied to many single-target search tasks. However, due to PSO's global optimization characteristics, it is easy to converge to a specific position in the search space and lose the ability to explore further. When faced with the problem of multitarget searching, it may become inefficient or even invalid. This paper proposes an Exploration Enhanced Robotic PSO (E2RPSO) method for multitarget searching problems for robotic swarms. The proposed method modifies the third item in the RPSO as an additional attraction term. This item not only enables the robot to avoid collisions but also guides the swarm to search unexplored regions as much as possible. This operation increases the swarm's task-specific (top-down) diversity, making the system cover a broader search area and avoid falling into local optimums. Besides, the aggregation degree and evolution speed factors are also included in determining the inertia weight of the proposed method, which adjusts the swarm's internal (bottom-up) diversity dynamically. The comparison results show that this method can balance the relationship between exploration and exploitation well, which has the potential to be applied to multitarget searching scenarios.

1. Introduction

As a physical system simulating the collaboration of swarm behaviors in biology, swarm robotics has been proved to have significant potential in military or civil applications, such as exploration of virgin territories [1], search and rescue [2], and cooperative transportations [3]. It aims to achieve emergent collective behaviors through the limited perception ability of swarm members and simple interactions with surroundings or teammates. The robot members in the system generally have the characteristics of low complexity [4]. At the system level, it uses redundancy to ensure robustness with excellent scalability and economy. This field has become a popular aspect of multirobot systems in recent years.

Swarm robotics is an entity realization of swarm intelligence in the domain of robotics. Many works apply the optimal searching ability of swarm intelligence (SI) algorithms to the cooperative search applications of robotic swarms [5]. In particular, the Particle Swarm Optimization (PSO) algorithm, as a prevalent optimization method, has been widely used [6, 7]. It can be traced to the work of Doctor et al. [8], in which the PSO algorithm is applied to the collaborative search application of swarm robotics for the first time. After that, many improvements have been proposed to improve the search performance in different aspects. For example, the application of PSO in a search and rescue task is simulated to locate a target using a robotic swarm [9]. An odor search task to determine the possible pollutants in the environment to be searched is also developed based on PSO [10].

It should be noted that the PSO algorithm can be applied to robotic applications in two ways. The first one is that the PSO algorithm can solve filtering problems or optimize

parameters for control [11, 12]. The other is that each particle in the algorithm can be mapped to a member robot in a swarm. The heuristic collaborative search can then be realized by imitating the algorithm's principles under the considerations of physical constraints, as shown in Table 1.

In particular, the Robotic PSO (RPSO) introduces an additional term into the standard PSO algorithm for obstacle avoidance [13]. In the same article, the authors also introduced another improved version of PSO (Darwinian PSO, DPSO) into the field of robots, known as RDPSO. It further adds an elimination mechanism based on Darwin's natural selection principles to increase swarm diversity. The RPSO and the RDPSO have been applied to some collaborative target search scenarios of swarm robotics [14, 15]. However, the additional term in those methods is only used for avoiding obstacles. The balance of exploration and exploitation still depends on the dynamic inertia weight, individual cognition (personal best), and social cognition (global best). The multitarget search problem requires the swarm to find as many targets as possible in the environment to be searched. These two methods are still suffering from premature convergence, i.e., not able to locate all the targets in an area. It is necessary to improve further the exploration ability of the swarm to meet the multitarget searching tasks.

This paper proposes a new robotic PSO-based approach named Exploration-Enhanced RPSO (E2RPSO). This method further improves the swarm diversity by adding an exploration enhancing operation, which is more suitable for multitarget search scenarios. The additional term in the original RPSO method is modified to maintain continuous searching states. Meanwhile, the adaptive dynamic inertia weight is still reserved to ensure the original performance. The simulation results show that the method proposed in this paper can find more targets than other PSO based approaches; i.e., it has better multitarget search performances.

The rest of this paper is organized as follows: Section 2 introduces related works. Section 3 states the multitarget searching problem and the assumptions, as well as the proposed E2RPSO method. The comparison results and discussions are presented in Sections 4 and 5. The conclusion is reached in Section 6.

2. Related Works

2.1. Swarm Intelligence and Swarm Robotics. Many Swarm Intelligence (SI) algorithms were directly used to design the collaborative behaviors of robotic swarms. Besides the PSO algorithm, other SI algorithms can be found for robotic swarm applications in the literature, such as Brain Storm Optimization (BSO) [16–18] and its extension in the field of multirobots named Brain Strom Robotics (BSR) [19]: Bees Algorithm (BA) [20, 21], Artificial Bee Colony (ABC) [22, 23], Ant Colony Optimization (ACO) [24, 25], Bacterial Foraging Optimization (BFO) [26, 27], Glowworm Swarm Optimization (GSO) [28, 29], Firefly Algorithm (FA) [30, 31], and Grey Wolf Optimizer (GWO) [10, 32], etc. The corresponding works are summarized in Table 2, which shows the original algorithm and related robotic applications correspondingly.

Table 1: Mapping PSO to robotic swarm searching tasks.

	PSO algorithm	Robotic swarm searching
Searching space	N-D space	2D or 3D space
Individuals	Particles	Robots
Constraints	Max. speed	Kinematics, dynamics, Local sensing
Anticollision	No	Yes
Position overlapping	Yes	No

2.2. PSO in Swarm Robotics. The PSO algorithm is derived from the study of the cooperative foraging behavior of birds. Its basic idea is to let a swarm of particles move in a solution space find a relatively optimal solution. The movements of the individuals are according to the individual information and shared knowledge from others. Each particle in the swarm has a position vector and a velocity vector. The fitness value of the current position is calculated according to the predefined objective function. The velocities and the positions of the particles will be updated iteratively. After a certain number of iterations, the particles will gradually tend to an optimal solution. Let the position vector of the particle i in an N-dimensional space be $\mathbf{X}_i = (x_{i1}, x_{i2}, \ldots, x_{iN})$, the corresponding velocity vector is $\mathbf{V}_i = (v_{i1}, v_{i2}, \ldots, v_{iN})$, the most popular version of the PSO algorithm is written as follows [33]:

$$\begin{cases} \mathbf{V}_i^{t+1} = w^t \mathbf{V}_i^t + c_1 r_1 \left(\mathbf{pb}_i^t - \mathbf{X}_i^t\right) + c_2 r_2 \left(\mathbf{gb}^t - \mathbf{X}_i^t\right), \\ \mathbf{X}_i^{t+1} = \mathbf{X}_i^t + \mathbf{V}_i^t, \end{cases} \tag{1}$$

where $\mathbf{V}_i^t$ and $\mathbf{X}_i^t$ are velocity and position vectors of particle i at time t, respectively, w^t is the dynamical inertia weight, $\mathbf{pb}_i$ is the personal best position of particle i, $\mathbf{gb}$ is the global best position of the swarm, c_1, c_2 are personal cognition and social congnition factors, respectively, and r_1, r_2 are random values in $[0, 1]$. Other parameters of PSO include population size n, maximum iteration number $G_{\max}$, maximum velocity $V_{\max}$, and so forth. It should be noted that the original version of the PSO algorithm does not have the inertia weight w^t, which weakens the ability of local search.

The introduction of inertia weight can dynamically adjust the weight according to the number of iterations or the swarm's current state in various versions of improved PSO algorithms. Thus the algorithm can balance the local search and global search ability, further enhancing the search efficiency and preventing the algorithm from falling into the local optimum prematurely.

The RPSO was proposed by taking obstacle avoidance into account for member robots [13]. Based on the original PSO algorithm, an additional item for obstacle avoidance is added; i.e., the first equation in (1) is changed to

$$\mathbf{V}_i^{t+1} = w_i^t \mathbf{V}_i^t + c_1 r_1 \left(\mathbf{pb}_i^t - \mathbf{X}_i^t\right) + c_2 r_2 \left(\mathbf{gb}^t - \mathbf{X}_i^t\right) + c_3 r_3 \left(\mathbf{po}_i^t - \mathbf{X}_i^t\right), \tag{2}$$

where $\mathbf{po}_i^t$ is the position of the anticollision relevant virtual attractive point, c_3 is the sensitivity of anticollision, and r_3 is

Table 2: SI algorithm inspired robotic applications.

Algorithms	Original Works	Robotic Applications
Particle swarm optimization, PSO	[6, 33]	[8, 13, 34]
Barin storm optimization, BSO	[16, 35]	[19]
Bees algorithm, BA	[20]	[21]
Artificial bee colony, ABC	[22]	[23]
Ant colony optimization, ACO	[24]	[25]
Bacterial foraging optimization, BFO	[26]	[27]
Glowworm swarm optimization, GSO	[28]	[29]
Firefly algorithm, FA	[30]	[31]
Grey wolf optimizer, GWO	[32]	[10]

a random number in $[0, 1]$. $\mathbf{po}_i^t$ can be determined by the obstacle and other members distributed in surroundings, which are generally perceived by equipped sensors.

Similarly, to avoid the premature problem, i.e., the robots getting stuck at the local optimum positions, some subsequent works were proposed to deal with this drawback. For example, some work combines the Darwinian PSO (DPSO) strategy, which adopts the nature selection principles to increase swarm diversity. Furthermore, to make them more physically implementable, the punishment and reward in DPSO were changed to the social exclusion and social inclusion operations of member robots. This method is called RDPSO [13]. The PSO algorithm's adaptive inertia weight is introduced into the RPSO method as well, such as Adaptive RPSO (A-RPSO) [14]. Based on the A-RPSO, some more work was proposed considering practical constraints such as relative localization, local sensing, limited communication, and the kinematic constraints [34]. Another work called ML-PSO combines a local search strategy with a modified PSO to achieve searching tasks that were presented in [36].

2.3. Multitarget Searching Problem. Generally, collaborative searching in swarm robotics can be divided into two main scenarios, depending on the number of targets to be searched: single target and multiple targets [5, 37]. For single target searching, the focus is on fusing the information collected by sensors equipped on robots and finally improving the accuracy of target position estimation. The multitargets scenario can be viewed as an extension of the single target case, where more uncertainties need to be considered [38]. For example, the number of targets may be unknown or even vary with time; the targets to be searched may be static or dynamic, etc. Thus the problem is how to design and implement a collaboration mechanism among the swarm members to find as many targets as possible efficiently under specific conditions.

To sum up, the PSO method is widely used in the collaborative search of robotic swarms. However, different from the swarm intelligence algorithm, it needs to consider more physical limitations of environments and member robots when mapping to swarm robotics. Moreover, when faced with multitarget search problems, task-related diversity must be added to improve the efficiency of regional coverage and find as many targets as possible. The contribution of the proposed method is to add this task-related diversity into the traditional RPSO for multitarget searching problems. The additional item in traditional RPSO is modified to have the ability of both anticollision and enhancing exploration. Details are as follows.

3. E2RPSO for Multitarget Searching

3.1. Assumptions. The following assumptions are made for the member robots, targets, and environments to make the research more practical.

(1) Robots. The mobile robot in this work has a holonomic model; i.e., it can move in any direction. The member robots have global positioning ability and know the boundary of the space to be searched, but the internal information of the search space is unknown. It can transmit information to the whole group through certain communication interactions. There are no central control and the global leader in the swarm.

(2) Targets. The targets will be treated as radio-beacons, which emit the nondirectional signals. The signals can affect the searching space with attenuation. The signal strength decreases as the distance increases. The signals broadcast by each target are indistinguishable and can be superimposed. The total number of targets is unknown; i.e., the swarm system needs to find as many targets as possible.

(3) Environments. The robotic swarm system will work in a 2D environment with obstacles; i.e., the member robots need to keep away from other members and the obstacles during the searching process.

(4) Fitness evaluation. The robot's fitness value at a certain location is the superposition of the strength of signals that can be sensed at that position.

(5) Target Handling. When the distance to a target is less than a certain value, the robot can recognize the target and handle it. Once the target is reached, its signal is no longer broadcast. Also, we assume that the robot that reaches the target first will stay at that position to handle the target, no longer participating in follow-up group actions. If more than one robot reaches the target position simultaneously, they will all stay there.

3.2. The Proposed Method. The velocity update strategy of the proposed E2RPSO can be represented as follows:

$$\mathbf{V}_i^{t+1} = w_i^t \mathbf{V}_i^t + c_1 r_1 \left[\mathbf{pb}_i^t - \mathbf{X}_i^t\right] + c_2 r_2 \left[\mathbf{gb}^t - \mathbf{X}_i^t\right] + c_3 r_3 \left[\mathbf{pa}_i^t - \mathbf{X}_i^t\right], \tag{3}$$

where the w, **pb**, **gb** are defined as same as the traditional PSO algorithm, $\mathbf{pa}_i^t$ is an additional attractive position. In the original RPSO approach, this item is used only for obstacle avoidance. The proposed approach uses it not only for anticollision with obstacles or other members but also for enhancing the exploration ability. The determination of the **pa**, c_3, and the dynamic inertia weight w^t are as follows.

3.3. Additional Attractive Positions. The additional attractive position can be determined by

$$\mathbf{pa}_i^t = \begin{cases} \mathbf{po}_i^t, & dn_i^t \le d_s, \\ \mathbf{pu}_i^t, & \text{Others}, \end{cases} \tag{4}$$

where dn represents the distance to the nearest obstacle or swarm-mate, d_s is the predefined safe distance, **po** is the position determined by the requirements of anticollision, **pu** is the exploration enhancement attractive position. In particular, **po** will be related to the range and bearing of the surrounding members or obstacles. Once a member robot detects obstacles or other members in a certain range, a virtual attraction point will be formed to attract the robot to avoid a possible collision, which can be written as follows:

$$\mathbf{po}_i^t = -(d_s \cos\theta_{\text{sol}}, d_s \sin\theta_{\text{sol}}), \quad \left(\mathrm{d}n_i^t \le d_s\right), \tag{5}$$

where θ_{sol} is the solution direction determined by obstacle avoidance techniques such as a modified polar vector field histogram (VFH) presented in [39]. To enhance the exploration ability, **pu** can be determined by the farthest unexplored region in the searching area, that is,

$$\mathbf{pu}_i^t = \text{argmax}_{\mathbf{p}\in\mathbf{P}_u^i} \left\|\mathbf{X}_i^t - \mathbf{p}\right\|, \tag{6}$$

where $\mathbf{P}_u^i$ is a set of locations in the searching area that the robot i has not visited.

3.4. Determination of c_3. Unlike that c_1 and c_2 are constants, here, c_3 is a variable that will be affected by the following listed different situations:

(1) *Collision Avoidance*. When a robot in the swarm is under the state of obstacle avoidance or anticollision, it is better to make the robot get rid of the collision danger as soon as possible. At this time, only the requirements of collision avoidance should be considered. Therefore, c_3 should be set far greater than c_1 and c_2, i.e., $c_3 \gg \max\{c_1, c_2\}$.

(2) *Improving Efficiency*. The value of c_3 in this paper will be determined according to the number of times a member robot repeatedly visits a certain area, denoted as m. At the beginning of the searching process, because most regions are not explored, the strategy should be more dependent on individual cognitive and social cognition coefficients, i.e., c_1 and c_2. With the execution of the search task, robots in the swarm may repeatedly access some regions that have been visited. Here we introduced a threshold σ. When the number of repeated visits to an area is greater than this threshold, the robot will increase the value of c_3 to enhance its exploration ability for the unknown regions.

The above strategy for determining c_3 can be summarized as follows:

$$c_3 = \begin{cases} A, & \mathrm{d}n_i^t \le d_s, \\ B, & \mathrm{d}n_i^t > d_s \text{ and } m < \sigma, \\ C, & \mathrm{d}n_i^t > d_s \text{ and } m \ge \sigma, \end{cases} \tag{7}$$

where $A, C \gg \max\{c_1, c_2\}$, $B \approx \max\{c_1, c_2\}$.

3.5. The Adaptive Inertia Weight. The value of dynamic weight w will be determined according to the evolution speed and aggregation degree of the population [14, 40, 41], which can be represented as follows:

$$w_i^t = w_0 - \alpha\left(1 - h_i^t\right) + \beta s_i^t, \tag{8}$$

where w_0 is the initial weight, $(1 - h_i^t)$ and s^t are evolutionary speed and aggregation degree, respectively, α and β in $[0, 1]$ are relevant coefficients. h_i^t can be written as follows:

$$h_i^t = \left|\frac{\min\left\{F\left(p\text{best}_i^{t-1}\right), F\left(p\text{best}_i^t\right)\right\}}{\max\left\{F\left(p\text{best}_i^{t-1}\right), F\left(p\text{best}_i^t\right)\right\}}\right|, \tag{9}$$

where $F(p\text{best}_i^t)$ is the best fitness value of the particle i till the time t. Obviously, $0 < h_i^t \le 1$, the larger the $(1 - h_i^t)$ is, the higher the speed of evolution, vice versa. The definitions of aggregation degrees are as follows:

$$s_i^t = \left|\frac{\min\left\{F^t(\text{best}), \overline{F^t}\right\}}{\max\left\{F^t(\text{best}), \overline{F^t}\right\}}\right|, \tag{10}$$

where $F^t(\text{best})$ is the best fitness value of the swarm in the time slot t, $\overline{F^t}$ is the average value of all particles in tth iteration. Also, $0 < s \le 1$, the larger value of s indicates the particles in the swarm are closer to each other. Since only the fitness values in the current iteration are involved in calculating s, it is more reflective of the swarm's current state.

4. Results and Discussion

We have conducted massive simulations to verify the effectiveness of the proposed method. These simulations are performed not only to validate the performance of the proposed approach but also to get comparisons with other PSO-based methods. The representative techniques are RPSO, RDPSO [13], and ML-PSO [36] mentioned above. The effects of population size, target number, and environment size on the corresponding methods are evaluated in different simulations. Details are as follows.

4.1. Configurations. The simulation configurations are shown in Figure 1, where the robot swarm, obstacles, and targets are illustrated. Except for experiments of evaluating the impact of map sizes, the targets and obstacles are distributed in a map of 1000×1000 units. The robots' initial position is in the middle of the left part of the environment. Robots can move maximumly two units in each iteration. The corresponding parameter configurations of the methods involved are all from the original works. For the parameter configurations not explicitly indicated in the original paper, the parameters consistent with the methods proposed in this paper are used. Specifically, we set $c_1 = 1$, $c_2 = 2$, for the proposed method, $A = 10000$, $B = 2$, $C = 20$. All the statistical results are the mean values after 30 independent tests.

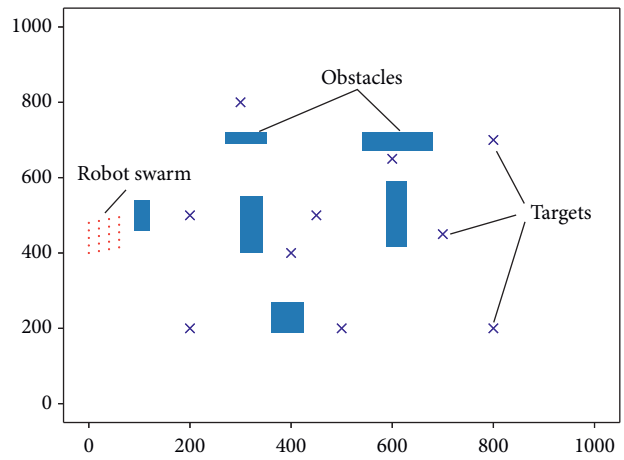

Figure 1: The simulation configuration.

4.2. E2RPSO Based Collaborative Searching

4.2.1. Single Target Search. A single target search process of the proposed method is shown in Figure 2. Ten robots are initialized at the middle-left of a 1000×1000 map with some obstacles and one target. By applying the proposed method, the target can be found through swarm cooperation. Generally, most of the PSO-based approaches can achieve the single-target search task. The above example shows that this method inherits the advantages of this kind of method, and the swarm can converge to the target position in the region to be searched.

4.2.2. Multitarget Search. A multitarget searching process of the proposed method is shown in Figure 3. Twenty robots are initialized at the middle-left of a 1000×1000 map with some obstacles. Ten targets are distributed randomly on the map. It can be seen that this method can find multiple targets to be searched in the environment and has corresponding effectiveness. Meanwhile, according to the preset settings, the robots that arrive at a target first will stay with and handle it. Not all robots will converge to the same position.

The exploration ability in the proposed method is enhanced. The unexplored area will have a higher probability of being searched. The results above indicate the effectiveness of the proposed method. Further, we also carried out quantitative evaluations of the proposed method and compared it with other PSO-based methods. The comparison methods involved in the simulation are the original Robotic PSO (RPSO), the Robotic Darwin PSO (RDPSO), and the Modified Local PSO (MLPSO). The impacts of the population size, the number of targets, and the environment are evaluated.

4.3. Impact of Population Size. The map configurations and target distributions are the same as in Figure 3. The swarm population is changed from 10 to 50, increasing by five each time. Each size has been tested 30 times independently. The average number of targets found by the four methods is shown in Figure 4. The proposed method can find almost all targets under each population of swarm size larger than 10. For the other compared methods, they are also able to find some targets in the environment. However, once the swarm converges to some target positions, the movement ability is insufficient, resulting in the inability to explore more areas.

The average iteration times before convergence of the compared methods are shown in Table 3. In general, for all compared approaches, the number of iterations before convergence decreases with the increase in population size. Because the map size is identical, with the swarm population size increasing, the algorithms have a higher probability of finding targets and convergence. It can be seen that the method proposed in this paper can not only find almost all targets but also have a lower average iteration time. The efficiency of this method is further proved.

We also calculated the extreme performance of all participating comparison methods, that is, the probability of finding all targets in all tests, as shown in Table 3. It can be seen that the performance of the proposed method is better than that of all other techniques involved in the comparison. Except for the minimal number of robots (10) and the extreme situation of target distribution (distributed in corners of the map), all targets can be found in other cases. For other methods, the possibility of finding more targets is proportional to the number of robots. However, the number of targets found by the compared methods are all less than the proposed method.

4.4. Impact of Targets Number. To further verify the multitarget search performance of this method, we tested the above four methods under different numbers of targets with the same map configuration (1000×1000). The robot population is fixed to 20, and the number of targets changes from 1 to 10. The average targets found by each compared method are shown in Figure 5. Each simulation has been tested 30 times independently.

Table 4 shows the probability of finding all targets in all tests and the performance of involved methods in the case of increasing targets. For the proposed approach, almost all targets can be found successfully in all tests except for several times when the target cannot be found in some extreme cases. In general, this method achieves the best multitargets search

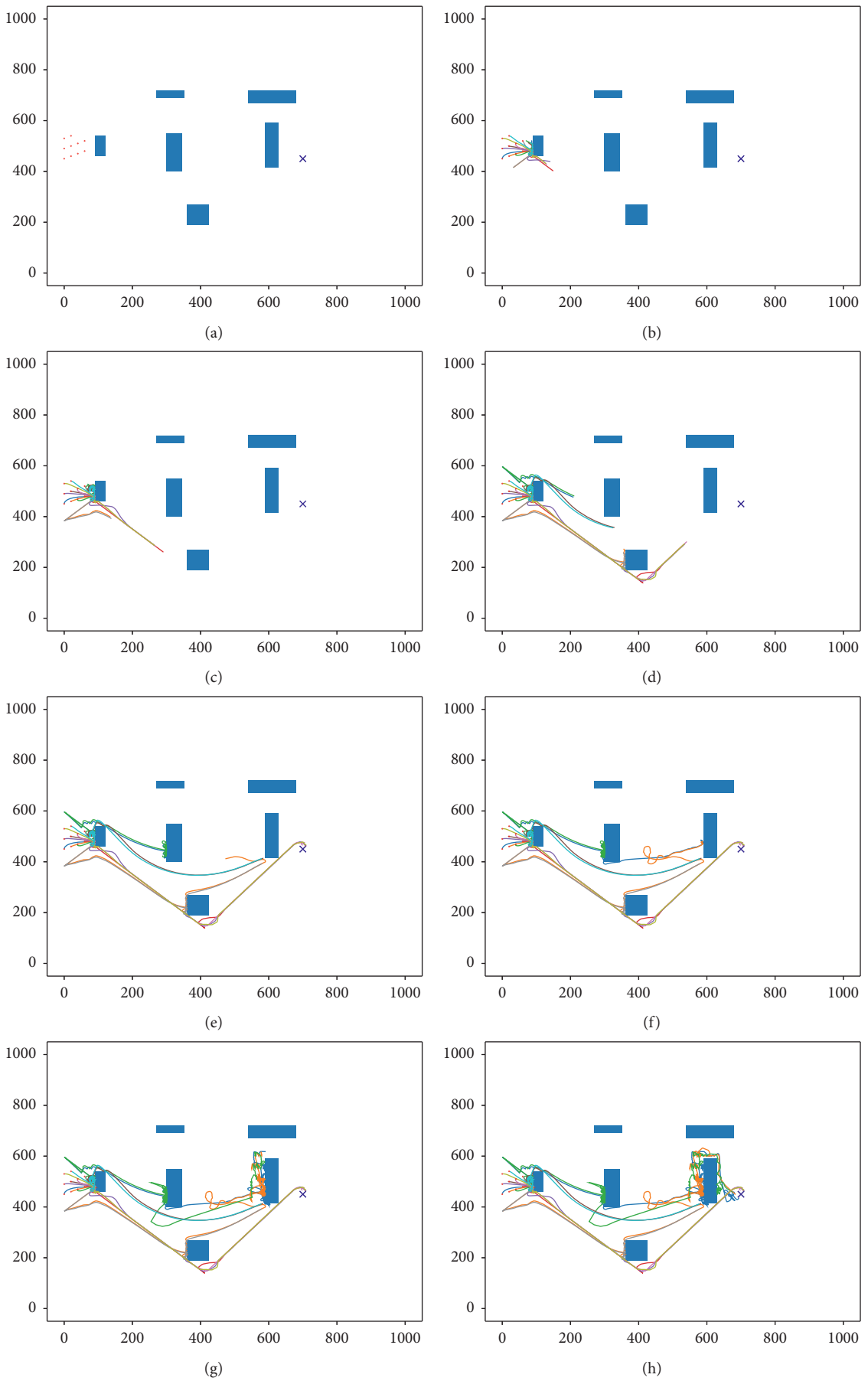

Figure 2: Single target search example of the proposed method. (a) $t = 0$. (b) $t = 100$. (c) $t = 200$. (d) $t = 300$. (e) $t = 400$. (f) $t = 500$. (g) $t = 800$. (h) $t = 1100$.

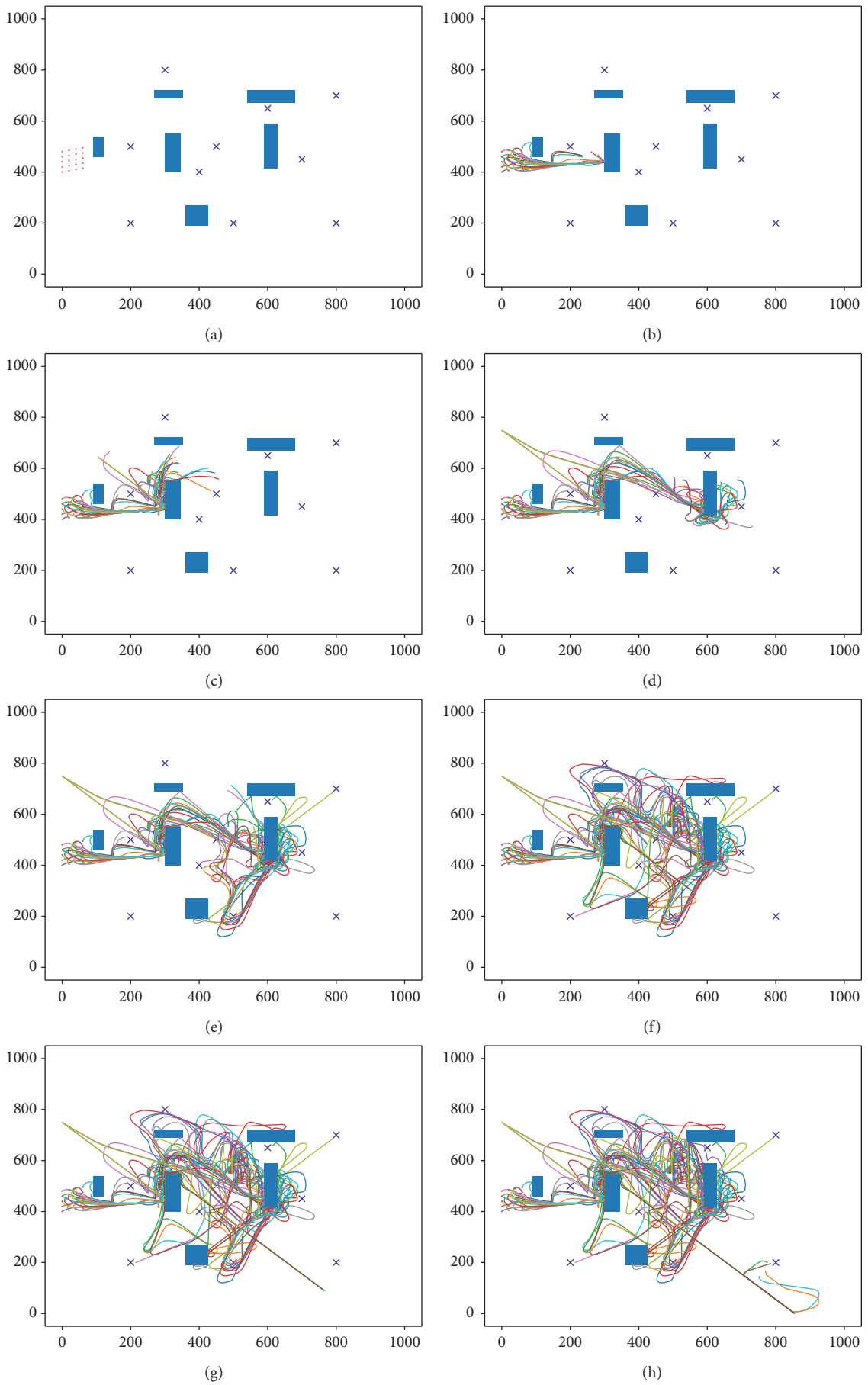

Figure 3: Multitarget search example of the proposed method. (a) $t = 0$. (b) $t = 100$. (c) $t = 200$. (d) $t = 400$. (e) $t = 800$. (f) $t = 1200$. (g) $t = 1400$. (h) $t = 1600$.

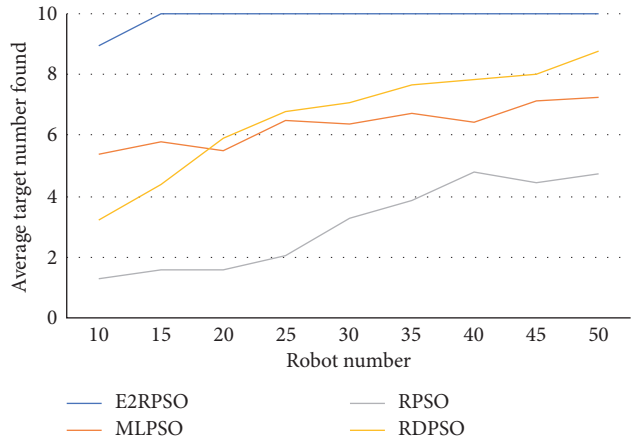

Figure 4: Comparison of average targets found under different population sizes.

Table 3: Iteration times and full success rates (%) under different population.

# of robots	E2RPSO		MLPSO		RPSO		RDPSO	
	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)
10	8943	3	9048	0	9747	0	9997	0
15	3538	100	4048	21	9436	0	9980	0
20	3073	100	4255	11	9341	0	9001	0
25	2161	100	3615	0	6513	12	2535	6
30	2219	100	3156	20	5172	12	3074	10
35	1958	100	3072	0	5781	0	2810	14
40	1836	97	3011	13	5623	22	2821	25
45	1625	100	3864	22	5575	25	2628	33
50	1430	100	2144	45	3867	30	2447	65

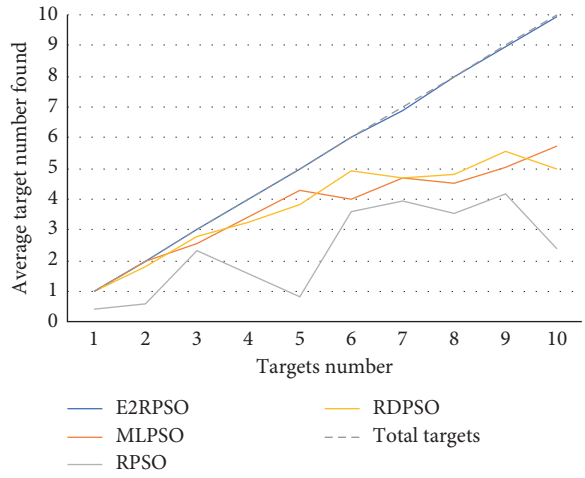

Figure 5: Comparison of average targets found under different targets number.

performance of all the techniques involved in the comparison. Furthermore, in terms of search time, it is approximately proportional to the target to be searched. The E2RPSO method proposed in this paper is similar to RDPSO and MLPSO in time consumption. RPSO takes a long time on average. Similarly, as shown in Table 4, the probability of finding all targets of the proposed method is significantly higher than other methods. Other methods participating in the comparison can find all the targets except when the number of targets is relatively small. With the increase in the number of targets, the possibility of finding all targets becomes smaller. However, the proposed method in this paper is less affected by the number of targets. The success rate of all targets that can be found was kept above 93% in all tests. Therefore, from this perspective, this method can still maintain a relatively low search time under the premise of finding more targets. It has the best search efficiency and performance over others.

4.5. Impact of Environment Size. We further evaluated the impact of environment size on the searching performance of all compared methods. The environment size increased from 500×500 to 2000×2000, and the target number is fixed at 10 with random distribution. The number of robots is set to 20, distributed in the middle of the left side of the map. The obstacles in the map are scaled proportionally. The tests are conducted 30 times independently under each map size. The average number of targets found is shown in Figure 6.

Figure 6 shows a comparison of the number of targets found by each method as the map grows. The proposed method is kept at a high level in all cases. Of course, it is not easy to find all targets with the increase of the map with a constant population size. But our method still shows superior performance in the tests. When the total number of targets is 10, the average value of the targets found is above 9.5 for all map sizes. Other methods have some effectiveness

Table 4: Iteration times and full success rate (%) under different targets number.

# of targets	E2RPSO		MLPSO		RPSO		RDPSO	
	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)
1	371	100	235	100	2226	100	365	100
2	743	100	403	100	3912	40	1013	87
3	860	100	930	67	4103	37	1130	83
4	1091	100	705	60	4565	3	2084	73
5	1662	97	815	50	4890	0	1271	37
6	2252	100	1653	47	4252	7	1866	37
7	2319	93	2514	27	6006	13	2519	13
8	2908	100	2054	10	5524	10	1537	3
9	2999	97	2364	0	3618	3	2765	7
10	3234	93	2677	0	5782	0	2653	0

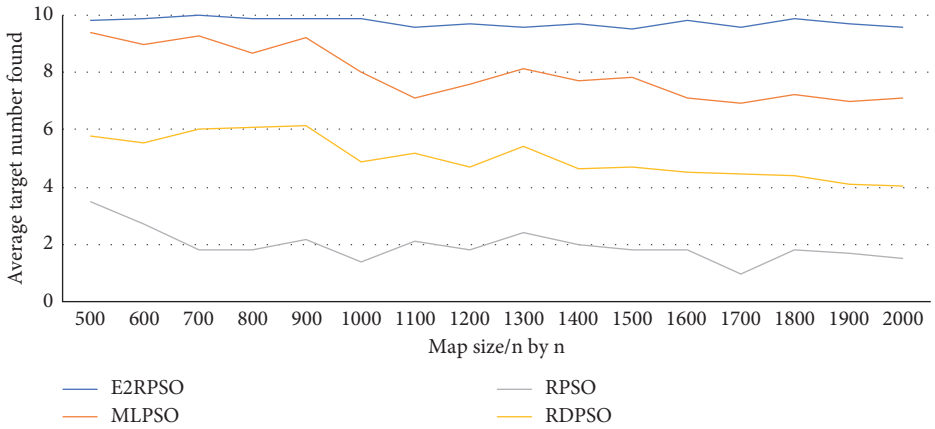

Figure 6: Comparison of average targets found under different environment size.

when the map has a small size. Still, with larger map sizes, the performance will be significantly affected. It should be noted that the MLPSO method also has the operation of enhancing the exploration ability of the unexplored area. The performance of this method is better than the other two methods.

From the iteration time shown in Table 5, RDPSO requires significantly more search time before convergence than E2RPSO and MLPSO in all cases due to the increasing operations of avoiding local optimal. The likelihood of finding all targets of RPSO is 0% in all cases. The iteration time always reached the preset maximum time of 10000. Compared with the MLPSO method, the E2RPSO method proposed in this paper is similar for iteration times, but the number of targets found by the proposed one is the largest. Similarly, we also evaluated each method's success rates that all the targets were detected in each environment size. The proposed method has a high probability of finding all the targets. When the scale of the map increase, the likelihood of finding all the targets decreases. However, it remains above 80%, as shown in Table 5.

5. Discussion

From the results above, we can see that the proposed method can collaboratively locate most of the targets in different cases. The PSO algorithm is initially inspired by the foraging behaviors from the biological swarm systems. It can search and locate targets in a region. However, due to the kinematical limitation of particles and the influence of inertia, it is not enough to adjust the population's diversity only by the personal cognition factor and the inertia weight, especially in need of regional coverage or multitarget search task. Through the extensive tests above, we found the following problems which need to be further discussed.

5.1. The Source of Diversity. Maintaining the diversity of a swarm is always an effective way to improve the performance of a swarm intelligence algorithm or application. The original PSO algorithm uses personal cognition and social cognition to reach the consensus with diversity, i.e., the social cognition for convergence and the personal cognition for diversity. However, such diversity among individuals will eventually be lost due to the unity of individual cognition and social cognition. A significant improvement of the follow-up work is the introduction of inertia weight (w^t). With the linear or adaptive inertia weight, the PSO algorithm can get more diversity and jump out of the local optimal. Different methods can balance the relationship between exploration and exploitation by adjusting the diversity of swarms.

The above two kinds of diversity come from the interior of the swarm, which we call bottom-up diversity. However, the diversity should also be related to the characteristics of different optimization problems or particular tasks of other swarm systems. This kind of diversity also conforms to the

Table 5: Iteration times and full success rate (%) under different map sizes.

Map size	E2RPSO		MLPSO		RPSO		RDPSO	
	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)	Iter.	FSR (%)
500 × 500	1537	93	2492	87	10000	0	3775	10
600 × 600	2289	100	3159	100	10000	0	5270	10
700 × 700	2705	100	3357	100	10000	0	4232	20
800 × 800	3181	100	3629	83	10000	0	5397	23
900 × 900	2976	97	3275	83	10000	0	4534	23
1000 × 1000	2913	100	3894	80	10000	0	5896	6.7
1100 × 1100	3012	100	3757	93	10000	0	6813	6.7
1200 × 1200	3379	93	3427	73	10000	0	4933	3.3
1300 × 1300	4170	100	3788	67	10000	0	7545	13
1400 × 1400	4153	97	4432	60	10000	0	10000	0
1500 × 1500	3901	90	5187	77	10000	0	6281	3.3
1600 × 1600	4673	93	6469	63	10000	0	6935	3.3
1700 × 1700	4996	93	6750	60	10000	0	8542	3.3
1800 × 1800	5019	83	6588	53	10000	0	8075	3.3
1900 × 1900	4906	83	6940	63	10000	0	10000	0
2000 × 2000	5049	80	7012	60	10000	0	10000	0

Darwinian evolution principle. To be different from the diversity mentioned above derived from the swarm itself, we turn this diversity derived from the environment into a top-down diversity. Swarm members may change some of their features according to the environment's changes or specific tasks, and then obtain the diversity from the external factors.

5.2. Multitarget Searching vs. Multimodal Optimization. According to the diversity discussion above, the proposed method added task-specific diversity into the original RPSO approach. Since the multitarget searching problem requires the searching procedure to locate the targets as much as possible in a particular region, more exploration ability is needed for the area coverage. The multitarget searching problem is similar to the multimodal optimization problem in the domain of optimization algorithms. Many multimodal objective functions have more than one optima. The multimodal optimization aims to locate multiple peaks/optima in a single run and maintain these found optima until the end of an optimization process. The strategies for those kinds of problems are also by enhancing the task-specific diversity such as Niching strategies, subpopulation strategies, etc.

Therefore, to solve specific optimization problems or specific swarm robotics tasks, it is impossible to complete the tasks ideally only by bottom-up diversity. Some tasks or environment-related diversity must be introduced, i.e., top-down diversity. These diversities may come from the prior knowledge of a problem or new cognitions of swarm members in performing tasks or solving problems. Such a situation exists in many actual biological communities, such as the foraging behavior of social organisms, which may be affected by the environment and lead to additional diversity. For example, in the collaborative foraging of ants, if some swarm members are disturbed by the surroundings, such as risk factors are detected, they may not continue to move according to the original pheromone mechanism but avoid the danger. They are thus introducing additional diversity, which may also provide more possibilities to find new foods.

6. Conclusion

This paper proposed an exploration-enhanced RPSO (E2RPSO) for collaborative multitarget searching in swarm robotics. The multitarget searching task requires the swarm system to locate as many targets as possible in a specific region; i.e., it requires the swarm to have the ability of additional exploration and area coverage. By keeping the requirements above in mind, we modified the third item in RPSO, which was originally only used for obstacle avoidance. By introducing this exploration enhancing operation, new task-specific diversity of the swarm was added, which we called "top-down" diversity. Besides, the adaptive dynamic inertia weight was used to ensure the "bottom-up" diversity of the original PSO. In the comparative simulation experiments, we evaluated the impacts of population size, target number, and environment size on each method. The results show that the method proposed in this paper can find more targets in a specific region without increasing the time cost. This method can balance the relationship between exploration and exploitation well and has excellent potential to be applied to practical multitarget searching scenarios.

Data Availability

The code used to support the findings of this study have been deposited in the GitHub repository (https://github.com/xrl2408/E2RPSO).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Jian Yang, Ruilin Xiong, and Xinhao Xiang contributed equally to this work.

Acknowledgments

This work was partially supported by National Key R&D Program of China under the Grant no. 2017YFC0804003, National Science Foundation of China (61761136008 and 61806119), Shenzhen Peacock Plan (KQTD2016112514355531), Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2017ZT07X386), the Science and Technology Innovation Committee Foundation of Shenzhen (JCYJ20200109141235597, ZDSYS201703031748284), Guangdong Provincial Key Laboratory (2020B121201001), and Special Funds for the Cultivation of Guangdong College Students Scientific and Technological Innovation ("Climbing Program" Special Funds, PDJH2020b0522).

References

[1] J. Yang, X. Wang, and P. Bauer, "Line and V-shape formation based distributed processing for robotic swarms," *Sensors*, vol. 18, no. 8, p. 2543, 2018.

[2] M. Bakhshipour, M. Jabbari Ghadi, and F. Namdari, "Swarm robotics search & rescue: a novel artificial intelligence-inspired optimization approach," *Applied Soft Computing*, vol. 57, pp. 708–726, 2017.

[3] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[4] J. Yang, X. Wang, and P. Bauer, "Formation forming based low-complexity swarms with distributed processing for decision making and resource allocation," in *Proceedings of the Control, Automation, Robotics and Vision (ICARCV), 2016 14th International Conference*, pp. 1–6, IEEE, Phuket, Thailand, November 2016.

[5] M. Senanayake, I. Senthooran, J. C. Barca, H. Chung, J. Kamruzzaman, and M. Murshed, "Search and tracking algorithms for swarms of robots: a survey," *Robotics and Autonomous Systems*, vol. 75, pp. 422–434, 2016.

[6] J. Kennedy and R. Eberhart, "Particle swarm optimization,"vol. 4, pp. 1942–1948, in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, November 1995.

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

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

[9] X. Songdong and Z. Jianchao, "Sense Limitedly, Interact locally: the control strategy for swarm robots search," in *Proceedings of the 2008 IEEE International Conference on Networking, Sensing and Control*, pp. 402–407, IEEE, Sanya, China, April 2008.

[10] U. Jain, R. Tiwari, and W. W. Godfrey, "Odor source localization by concatenating particle swarm optimization and Grey wolf optimizer," in *Advanced Computational and Communication Paradigms*, pp. 145–153, Springer, Cham, Switzerland, 2018.

[11] G. A. R. Ibraheem, A. T. Azar, I. K. Ibraheem, and A. J. Humaidi, "A novel design of a Neural Network-based fractional PID controller for mobile robots using hybridized fruit fly and particle swarm optimization," *Complexity*, vol. 2020, Article ID 3067024, , 2020.

[12] G. Gao, F. Liu, H. San, X. Wu, and W. Wang, "Hybrid optimal kinematic parameter identification for an industrial robot based on BPNN-PSO," *Complexity*, vol. 2018, Article ID 4258676, , 2018.

[13] M. S. Couceiro, R. P. Rocha, and N. M. Ferreira, "A novel multi-robot exploration approach based on particle swarm optimization algorithms," in *Proceedings of the Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium*, pp. 327–332, IEEE, Kyoto, Japan, November 2011.

[14] M. Dadgar, S. Jafari, and A. Hamzeh, "A PSO-based multi-robot cooperation method for target searching in unknown environments," *Neurocomputing*, vol. 177, pp. 62–74, 2016.

[15] A. S. Kumar, G. Manikutty, R. R. Bhavani, and M. S. Couceiro, "Search and rescue operations using robotic darwinian particle swarm optimization," in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 1839–1843, IEEE, Udupi, India, September 2017.

[16] Y. Shi, "An optimization algorithm based on brainstorming process," *International Journal of Swarm Intelligence Research*, vol. 2, no. 4, pp. 35–62, 2011.

[17] L. Ma, S. Cheng, and Y. Shi, "Enhancing learning efficiency of brain storm optimization via orthogonal learning," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[18] L. Qu, Q. Duan, J. Yang et al., "Brain storm optimization algorithm with cooperative learning strategy," in *Proceedings of the International Conference on Swarm Intelligence*, pp. 243–250, Springer, Rome, Italy, October 2020.

[19] J. Yang, Y. Shen, and Y. Shi, "Brain storm robotics: an automatic design framework for multi-robot systems," in *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, Glasgow, UK, July 2020.

[20] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm—a novel tool for complex optimisation problems," in *Intelligent Production Machines and Systems*, pp. 454–459, Elsevier Science Ltd, Amsterdam, The Netherlands, 2006.

[21] A. Jevtić, P. Gazi, D. Andina, and M. Jamshidi, "Building a swarm of robotic bees," in *Proceedings of the 2010 World Automation Congress*, pp. 1–6, IEEE, Kobe, Japan, September 2010.

[22] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, Kayseri, Turkey, 2005.

[23] A. Banharnsakun, T. Achalakul, and R. C. Batra, "Target finding and obstacle avoidance algorithm for microrobot swarms," in *Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1610–1615, IEEE, Seoul, South Korea, October 2012.

[24] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137–172, 1999.

[25] N. R. Hoff, A. Sagoff, R. J. Wood, and R. Nagpal, "Two foraging algorithms for robot swarms using only local communication," in *Proceedings of the 2010 IEEE International Conference on Robotics and Biomimetics*, pp. 123–130, IEEE, Tianjin, China, December 2010.

[26] S. Das, A. Biswas, S. Dasgupta, and A. Abraham, "Bacterial foraging optimization algorithm: theoretical foundations,

analysis, and applications," *Foundations of Computational Intelligence*, vol. 3, pp. 23–55, 2009.

[27] B. Yang, Y. Ding, and K. Hao, "Target searching and trapping for swarm robots with modified bacterial foraging optimization algorithm," in *Proceeding of the 11th World Congress on Intelligent Control and Automation*, pp. 1348–1353, IEEE, Shenyang, China, June 2014.

[28] K. N. Krishnanand and D. Ghose, "Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications," *Multiagent and Grid Systems*, vol. 2, no. 3, pp. 209–222, 2006.

[29] K. Krishnanand and D. Ghose, "Detection of multiple source locations using a glowworm metaphor with applications to collective robotics," in *Proceedings of the Swarm Intelligence Symposium, 2005. SIS 2005*, pp. 84–91, IEEE, Pasadena, CA, USA, 2005.

[30] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Proceedings of the International Symposium on Stochastic Algorithms*, pp. 169–178, Springer, Sapporo, Japan, October 2009.

[31] P. Nunzia and S. Marano, "Discrete firefly algorithm for recruiting task in a swarm of robots," in *Nature-Inspired Computation in Engineering*, pp. 133–150, Springer, Cham, Switzerland, 2016.

[32] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[33] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *in Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence*, pp. 69–73, IEEE, Anchorage, AK, USA, May 1998.

[34] J. Yang, X. Wang, and P. Bauer, "Extended PSO based collaborative searching for robotic swarms with practical constraints," *IEEE Access*, vol. 7, pp. 76328–76341, 2019.

[35] Y. Shi, "Brain storm optimization algorithm in objective space," in *Proceedings of the 2015 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1227–1234, IEEE, Sendai, Japan, May 2015.

[36] M. N. Rastgoo, B. Nakisa, and M. Z. Ahmad Nazri, "A hybrid of modified PSO and local search on a multi-robot search system," *International Journal of Advanced Robotic Systems*, vol. 12, no. 7, p. 86, 2015.

[37] J. Li and Y. Tan, "A two-stage imitation learning framework for the multitarget search problem in swarm robotics," *Neurocomputing*, vol. 334, pp. 249–264, 2019.

[38] J. Li and Y. Tan, "A probabilistic finite state machine based strategy for multitarget search using swarm robotics," *Applied Soft Computing*, vol. 77, pp. 467–483, 2019.

[39] J. Yang, X. Wang, and P. Bauer, "V-shaped formation control for robotic swarms constrained by field of view," *Applied Sciences*, vol. 8, no. 11, p. 2120, 2018.

[40] X. Yang, J. Yuan, J. Yuan, and H. Mao, "A modified particle swarm optimizer with dynamic adaptation," *Applied Mathematics and Computation*, vol. 189, no. 2, pp. 1205–1213, 2007.

[41] Y. Shen, J. Yang, S. Cheng, and Y. Shi, "BSO-AL: brain storm optimization algorithm with adaptive learning strategy," in *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–7, IEEE, Glasgow, UK, July 2020.