2014 IEEE International Conference on Systems, Man, and Cybernetics
October 5-8, 2014, San Diego, CA, USA

Improved Group Explosion Strategy for Searching Multiple Targets using Swarm Robotics

Zhongyang Zheng, Jie Li, Junzhi Li and Ying Tan

Abstract— **In this paper, an improved group explosion strategy (IGES) is proposed for searching multiple targets using a swarm of simple robots. The strategy is based on our previous work which has several shortcoming especially when number of targets and robots are large. IGES is simple and fast with great adaptability and only one parameter. The simulation results demonstrate that IGES has great efficiency in all aspects including searching time, energy consumption and computation overload. IGES also shows great stability and adaptiveness in both small and large scale problems.**

I. Introduction

Swarm robotics has achieved significant progress benefiting from the development of artificial intelligent [1]. Swarm robotics can be used in many applications, especially those require large amount of robots and time as well as difficult [2] or dangerous [3] for human beings, e.g. foraging, surveillance, monitoring and search-and-rescue. These applications can be abstracted as a multiple target searching problem with restrictions in the environment, such as obstacles. Searching strategies for solving this abstracted problem can be adopted to many applications and remains an important task for swarm robotics researchers.

Inspired from the swarm intelligence, most of swarm robotics searching problems use fitness values to guide the robots in the swarm. In both simulation and entity researches, these values can determine the distance of the targets from the robot. Fitness values usually have some corresponding meanings in physical world, such as Euclidean distance [4], chemical clues [5] or some type of potential functions [6]. Fitness generated in these ways are continuous and as regular as contours. Such problems can be solved with gradient descent methods. However, hardware designs in swarm robotics should be as simple as possible leading to low quality on-board sensors and fault sensing results and errors.

To make the problem more realistic and challenging, discrete fitness values are introduced into the problem. After rounding continuous sensing result into discrete values, the sensing errors can reduce significantly to compensate for the cheap hardware. The discrete fitness makes the problem a little more difficult yet local searching strategies can still solve the problem without much effort.

Robots in the swarm are very small, cheap and simple. Robots should have as limited abilities as possible including, but not limited to, motion, storage, sensing, communication, energy consumption and computation. In this way, software of the swarm plays a very important role in emerging cooperating behaviors from the swarm. Thanks to the similarity in problem, many swarm intelligence algorithms and their variants are used for cooperative strategy of swarm robots, including PSO [7], ACO [8] and etc. However, these algorithms may not fit for many restrictions in swarm robotics, such as limited sensing and communication ranges, dynamic adding or removing the robots and continuous movement.

In our previous work [9], a swarm robotics searching strategy inspired from the explosion phenomenon in nature was proposed. In this group explosion strategy (GES), the entire swarm is dynamically divided into several groups which search for the targets in parallel. Although GES shows good performance in the searching problem with regular fitness, the strategy still remains several shortcomings which may lead to inconvenience of a group during the searching problem. Based on the same explosion inspiration, which proved to be useful in GES, an improved GES (IGES) is proposed in this paper. IGES is more simpler yet shows much better performance than GES in our simulation. The searching problem is first introduced in Section II. Then section III analyses shortcomings of GES and describes IGES in detail. Experimental results and discussions are presented in IV. Finally, Section V concludes work in this paper.

II. Problem Statement

In our previous work, the problem for searching multiple targets is introduced. The multiple target searching problem proposed in our previous work shares the basic idea from other searching problems, e.g. [10], [11], except the fitness values in the environment are discrete. The reason for discrete fitness values is that robots in the swarm robotic researches should be as simple as possible. The on-board sensors should be simple and cheap, and are thus inevitably inadequate for sensing the targets. The direct result from the sensors may be continuous. However, considering the errors, we divide the whole sensing range into several intervals and approximate the result to the nearest interval. The results thus become discrete yet more reliable than the raw values.

Detail of the problem is defined as follows. There exists m targets in the environment and n robots search for the targets with the aid of fitness values by the targets. Fitness values are inversely proportional to the distance from the target, as shown in Figure 1. The swarm should search and collect the targets as quickly as possible. A target requires 10 iteration*robot to be collected.Cooperation of multiple

All authors are with Key Laboratory of Machine Perception and Intelligence (Peking University), Ministry of Education; Department of Machine Intelligence, School of Electronics Engineering and Computer Science, Peking University, Beijing, China, 100871.

This work was supported by the National Natural Science Foundation of China with grants no. 61375119, 61170057 and 60875080. Professor Ying Tan is the corresponding author.

978-1-4799-3840-7/14/$31.00 ©2014 IEEE

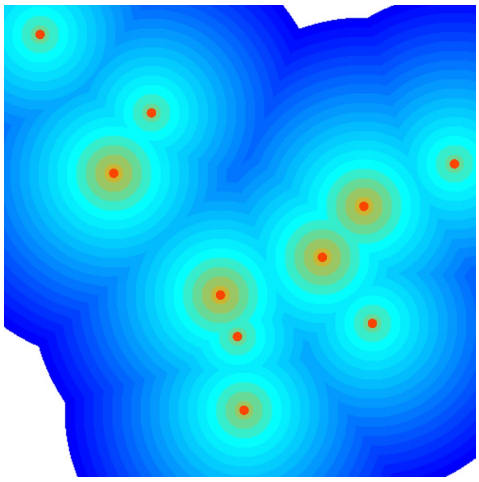


Fig. 1: Screenshots of the searching problem at the beginning of the simulation. Red circles stand for the targets. The background color illustrates fitness of that position. Position of targets are generated randomly. Robots are not illustrated in this figure.

individuals can shorten the time of collecting one target. A collected target disappears as well as its fitness.

Targets shape as circles or spheres with a radius of $Size_t$. A target is found only if a robot run into it and the robot can start collecting at its position. Fitness value of the target itself is generated randomly, ranges from $F_{Max}-2$ to F_{Max}. $Size_t$ and F_{Max} are set to 10 and 20 respectively in this paper.

The n robots in this problem are autonomous and mobile, designed to be as simple as possible. They are modeled as squares or cubes with the ability to move freely in environment. The swarm has no leader nor unique IDs and shares no common coordinate systems nor global position systems. Each robot can sense the fitness at its current position and detect the relative positions of the neighbour robots within limited sensing range. Each robot communicates explicitly with others only for sharing current fitness to neighbours. Each individual executes the same algorithm but acts independently and asynchronously from all others.

The sensing range of neighbour robots is set as $4Size_t$ so that robots can detect nearby neighbours with different fitness for better cooperation. All the detected positions are relative which can be done through infrared sensors and angle transducers. Therefore, direct communications or shared position system is not necessary. If F_{Max} is quite small, robots can detect neighbors' fitness through colored on-board lights. Otherwise, just like the situation in this paper, robots share their fitness values to all their neighbors through direct communications or other strategies which is not focused in this paper.

Robots have a speed limit of $2Size_t$ per iteration so that they can past at most one fitness level in one iteration and react quickly to the fitness. Individuals can also maintain 10 of their past states including position and the corresponding fitness. Positions in history are relative to the local coordinating system and updated according to robots' movements. Past states cannot be shared among the swarm since complex communications and localizations are required. This conflicts with the principle of the swarm robotics: simple and elegant [12].

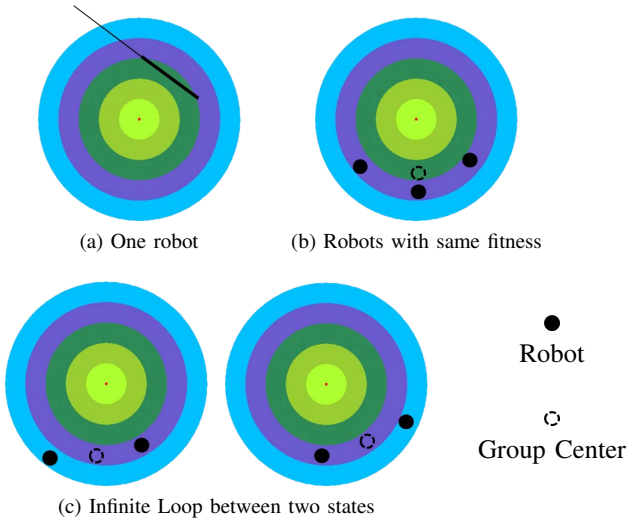
III. Improved Group Explosion Strategy

In previous work, a Group Explosion Strategy (GES) is proposed to solve the original searching problem. GES outperforms the comparison algorithm, yet still cannot perform well in certain circumstances. In this paper, an Improved GES (IGES) is proposed based on the same idea of inter and intra group cooperation, yet with simpler strategy, fewer parameters and better performances. In this section, we first analyze the shortcoming of the GES briefly, then describe the strategies in IGES and finally briefly prove that IGES can converge.

A. Shortcoming of the Group Explosion Strategy

GES shows quite good performance than the baseline algorithm especially when the fitness is not very adequate in the environment. The swarm shows quite good cooperation when population is not very large. However, as shown in the results in Section IV-D, when the population grows to more than certain extent, i.e. the swarm size may be crowded for the problem, performance of GES may become worse than the comparison algorithm. This means the cooperation of GES does not take full advantage of such a large population and therefore improvements are proposed in this paper.

The brief idea of intra-group cooperation in GES is that the group moves the center towards the best individual in the group. If multiple individual shares the same fitness, a random one is picked. However, robots may get stuck or fall back to places with worse fitness in certain cases, such as the three situations shown in Figure 2.



(a) One robot (b) Robots with same fitness

(c) Infinite Loop between two states

Fig. 2: Three situation which GES does not perform well.

In Figure 2a when only one robot is in the group, the robot may bounce along the bold black line, since all of the best

positions of the robot in history are on this line and the robot can hardly get out of the line. The robots may stuck in this situation for tens or hundreds of iterations before another robot run into its sensing range to save it.

In Figure 2b when several robots shares the same fitness, the group center may have a better fitness, but when the group follows the strategy moves its center towards a robot (say the bottom one), the whole group moves in the opposite of the target. Although the swarm can get back in a few iterations and hopefully select another robot to move the center to, the efficiency can be improved with better strategy.

Finally, in Figure 2c, an infinite loop may occur for the situation of two robots with different fitness. The center moves back and force between the two positions in the better fitness area and repeats the two states shown in the figure. Solutions for these three cases will be illustrated in the proposed IGES.

B. Improved Group Explosion Strategy

The motivation of GES is to utilize both inter and intra group cooperations during the search. From observation in our simulation program, cooperations within the group occur much more than that among groups, since the environment is very large compared to the size of the robots and their sensing areas. In this way, we focus on intra-group cooperations in IGES. Robots adopts different strategies based on their current states. Strategies in IGES are simpler compared with GES, yet show better performance. This indicates the improvements in IGES are critical.

Similar in GES, a robot and all its neighbours are denoted as a group. The strategy of a robot in IGES is selected first based on the size of the group: multiple robots or single robot; then based on the fitness values in the group (multiple robots condition) or the history states of the robot (single robot condition). There are four strategies in IGES as shown in Table I. Different conditions may share the same strategy.

The final velocity and position update functions for robot i are shown below:

$$V_i(t) = S_i(t) + R_C \cdot R_p \tag{1}$$

$$P_i(t) = P_i(t-1) + \frac{V_i(t)}{\|V_i(t)\|} \times 2Size_t \tag{2}$$

where $S_i(t)$ is the velocity update vector from the strategy it adopted, R_C is the factor shown in Table I and R_p is a unit random vector. The introduction of R_C is to avoid the shortcoming mentioned above and it's discussed in the following section. $2Size_t$ is the maximum speed limit mentioned in the problem statement. Velocities are normalized to the maximum speed.

Since the diameter of the sensing area is the same as radius of the fitness annulus, the maximum difference of fitness value within a group is 1. Therefore, in multiple robot condition, there are only two possibilities: all robots in the group have same fitness or not. Therefore, two strategies (No. 1 and 2) are used for different situations. In GES, we split the group when the group size exceeds certain limit β_G and this strategy is kept yet simplified in IGES: a vector leaving the group center is added to the velocity. Since the split strategy is same with the strategy with same fitness (No. 1), therefore splitting is only considered when fitness values are different (1+2 in in Table I).

TABLE I: Brief Summary of the IGES

Group Size	Fitness Condition	Strategy	R_C
$\geq \beta_G$	Different Fitness	No. 1+2	
$\in [2, \beta_G)$	Different Fitness	No. 2	$^1/_{10}$
≥ 2	Same Fitness	No. 1	
$= 1$	Best in history	No. 3	0
	Worse than last time	No. 4	1
	Better history in the earlier	No. 4	$^1/_{10}$

In single robot condition, the history states are used to select strategy for the robot. History states here include all the past states stored (at most 10) and the current state of the robot. Considering the current fitness and fitness values in history, three situations may happen: 1) the current fitness is the best fitness values in history, although maybe some other history states share the same fitness; 2) the fitness is worse than last iteration; 3) there are better fitness in a few iterations ago. The last two situation shares the same strategy, but with different R_C so as to avoid shortcoming shown in Figure 2a.

1) Strategy 1: This strategy is used for both splitting groups and multiple robots with same fitness. The robots in the group leave the group center as shown in (3).

$$S_i(t) = P_i(t) - \frac{\sum_{j \in N_i(t)} P_j(t)}{\|N_i(t)\|} \tag{3}$$

where $P_j(t)$ is the position of robot j at time t and $N_i(t)$ is the collection consists of all robots in the group of robot i.

2) Strategy 2: This strategy is used for multiple robots with different fitness. The strategy moves the center of the group towards the center of maximum fitness positions in the group.

$$S_i(t) = \frac{\sum_{j \in \hat{N}_i(t)} P_j(t)}{\left\|\hat{N}_i(t)\right\|} - \frac{\sum_{j \in N_i(t)} P_j(t)}{\|N_i(t)\|} \tag{4}$$

where $\hat{N}_i(t)$ is the sub collection of $N_i(t)$ containing all the robots with the maximum fitness value.

3) Strategy 3: This strategy happens when the robot is currently the best in history. Therefore, the robot just continues with its previous search direction, even other positions with same fitness exist in history.

$$S_i(t) = V_i(t-1) \tag{5}$$

4) Strategy 4: This strategy is for situation when there are better fitness in history. The robot moves towards the center of the positions with best fitness.

$$S_i(t) = \frac{\sum_{p \in \hat{H}_i(t)} p}{\left\|\hat{H}_i(t)\right\|} - P_i(t) \tag{6}$$

where $\hat{H}_i(t)$ is the collection of positions with the best fitness value in robot i's history.

IV. Simulation Results and Discussions

To illustrate the performance of the IGES strategy proposed in this paper, we compare it with the previous GES and another searching algorithm inspired form PSO. Several experiments and discussions are conducted in this section. The algorithms are simulated in our self-built simulation platform [13] and tested under various environment setups to see its adaptability. In the first validation experiment, IGES is tested in the original problem with default setup to see if it's capable for solving the problem. Parameter analysis of the IGES is shown in the second experiment. In the third scalability experiment, algorithms are tested in environments with various numbers of robots and targets in larger scale.

In most of the experiments, the simulations stop when all the targets are collected and the essential criteria for judging the performance is the iteration used. Iteration determines how fast the swarm can collect the targets and shorter iteration indicates the better performance.

All the experiments in this section use the same basic setups. The map size is 1000*1000 while the size of a robot is 1 and its sensing range is 20. It is very difficult a robot to search by itself, therefore cooperation is very important in the problem. In each test, 20 randomly generated maps are used for each setup and each method is repeated 20 times. Average results of these 400 runs are presented.

In this paper, parameters of the two comparison algorithms remains the same as tuned in previous work. The IGES has only one parameter β_G and shares the same value in GES. So no parameter needs to be tuned. In Section IV-C, we will show the trend of this parameter is quite the same as in GES.

A. Comparison Algorithms

Besides GES, another comparison algorithm used in this paper is RPSO [10]. In RPSO, each robot acts as a particle in PSO and the swarm adopts spacial-based topology for calculating gbests. RPSO may also suffer from vibration in the experiments, since the sensing range is quite small compared to the environment. In such case, a small random unit vector R_P is introduced if both pbest and gbest are the current position.

For better comparison, all three algorithms in the experiment share same strategy of obstacle avoiding and history state updating. Therefore, the difference of the strategy can be shown in the results directly.

B. Validation Experiment

Validation results are shown in Table II. Results of three algorithms are presented only when the number of targets and robots are 10, 30 and 50. The full results of different setups are shown in scalability experiment. Results after collecting 50% and 100% targets are shown in the table. In the table, m indicates number of targets and n is population.

The column "Collect" indicates the current collected targets after collecting 50% targets. In certain cases, two targets may be collected at the same time, so the value is slightly more than half of m. "Distance" determines the total distance of all robots for searching all the targets. In real robots, moving is one of the most energy consuming behavior and the energy consumed is related to the total distance. The final column shows the cpu time used for all the robots in each iteration which is the time of the algorithm itself excluding environment updates and result outputs. Calculation is another energy consuming activity in real robots, therefore a shorter CPU time saves the swarm for a smaller battery.

TABLE II: Validation results.

m	n	Algo	50% Collect	50% Iteration	Distance	100% Iteration	CPU Time /Iteration
10	10	GES	5.01	190.35	35414.48	377.10	7.43
		IGES	**5.03**	**121.03**	**25268.52**	**265.47**	**4.24**
		RPSO	5.01	177.29	32453.19	420.29	6.81
	30	GES	**5.03**	161.14	73738.18	266.80	19.45
		IGES	5.02	**91.41**	**50632.52**	**173.80**	**9.76**
		RPSO	5.02	113.97	60579.84	240.12	14.50
	50	GES	**5.04**	157.79	114149.20	250.37	37.98
		IGES	5.02	**83.15**	**73786.94**	**150.85**	**13.16**
		RPSO	**5.04**	99.20	86979.09	200.69	20.61
30	10	GES	15.01	312.39	62465.30	678.63	36.82
		IGES	**15.03**	**205.21**	**42229.20**	**458.57**	**24.46**
		RPSO	15.01	340.95	58900.61	799.04	40.05
	30	GES	**15.05**	232.41	112428.00	411.64	40.72
		IGES	**15.05**	**134.84**	**81650.54**	**286.52**	**23.58**
		RPSO	15.03	219.66	113554.00	460.07	43.19
	50	GES	15.06	230.35	167586.90	370.74	56.09
		IGES	**15.07**	**112.42**	**112432.30**	**234.27**	**23.27**
		RPSO	15.06	190.39	161752.90	380.59	48.91
50	10	GES	**25.05**	403.89	81050.37	893.69	39.46
		IGES	25.04	**257.70**	**53126.18**	**590.53**	**23.66**
		RPSO	25.01	460.27	76870.97	1068.99	44.37
	30	GES	25.07	271.16	138399.10	511.28	41.97
		IGES	**25.10**	**157.91**	**99513.25**	**355.39**	**24.00**
		RPSO	25.03	275.19	142912.20	589.33	47.09
	50	GES	**25.10**	270.03	201971.60	449.70	60.96
		IGES	**25.10**	**128.35**	**136241.00**	**288.10**	**24.88**
		RPSO	25.05	236.76	206335.30	491.28	58.79

In the table, IGES dominates in almost all the columns except "Collect". The strategy in IGES is effective and the improvements proposed in this paper are significant.

GES uses the most iterations for collecting 50% of the targets yet RPSO has the worst iteration performance for all targets. This shows the same trend in our previous work which shows shortcomings of the cooperation in GES. The iteration results of IGES beat both the comparison algorithms for at least 30%. IGES solves the shortcomings in GES and as a sequence boosts in the performance to a great extent. IGES also shows very stable performance in collecting both 50% and 100% targets, as it uses about 220% iterations when collecting all targets than that of half targets. This is very reasonable as the collecting becomes difficult as the remain targets reduces. The cooperation strategy in IGES shows great performance and adaptability in different target distributions and densities.

The distance results in the table are the total moving distance of all robots in all iterations. Therefore, IGES have shorter distance thanks partly to the low iteration. If we

divide the distance by the iteration used, GES and IGES shares almost the same moving distance, as robots moves at the maximum speed in most of times in both strategies. In RPSO, length of each step may differ for different situations and the average distance is 10% shorter than GES and IGES which is one of the reasons that it takes more iterations than GES for collecting 100% targets.

The CPU time can be used to justify the simpleness of the swarm robotics algorithm. The main idea of the swarm robotics is simple yet massive. Hardware design of the swarm can benefit from a simple strategy which saves cost and resources. IGES has the most quick CPU time among three algorithms, 10-30% and 25-50% quicker than RPSO and GES. The improvement of IGES simplifies the strategy while shows better performance. This shows that a simple strategy with the help of cooperation can have promising performance which is exactly the fundamental ida of swarm robotics.

C. Parameter Analysis

The only parameter in IGES is the group size limit β_G inherited from the GES. The meaning of the parameter remains unchanged in the IGES and thus we suggest the original parameter in GES should work and at least close to the best value. Therefore we use the original value in IGES instead of tuning the value in advance in the experiment. We now verify this hypothesis in experiment. From the results in this section, this very value shows great performance among the entire value fields.

The testing environment is chosen with middle size of number of targets and population size, i.e 30 for both of them. Since the population is 30, the maximum values for the parameter is 30 and the minimum is 4. The experiment results are shown in Figure 3. Only iteration results are shown in the figure as it's the most important criteria and should be used for parameter tuning.

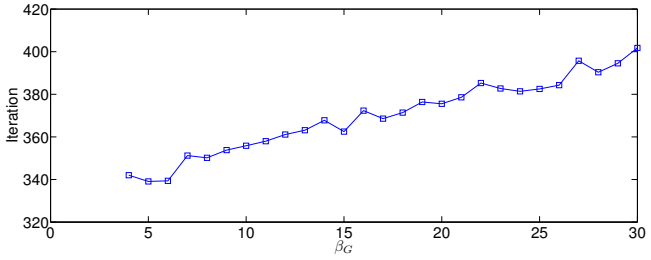


Fig. 3: Parameter analysis result of β_G.

The trend of the parameter is quite similar with that of GES. This indicates that the basic idea of the two strategies remains unchanged and the parameter is playing the same role in the two strategies. The original value of β_G in GES is 5 which is still the best position in the figure. This proves the hypothesis above is correct.

The influence of the parameter is obvious and shows a significant effect on iteration performance. The best value is 15% quicker than the worst value. The best value is reached when β_G is 5 or 6 and the performance becomes worse as β_G increases. Excluding the vibration from randomness in the strategy, the iteration is almost linear to the β_G value when it exceeds 6. This is easy to understand that the cooperation strategy works best when group size is 5 or 6. When group size is smaller, the group may converge slower. If the group size is large, the additional robots in the group will not accelerate the intra group cooperation but reduces the parallelism of the entire swarm. Therefore, the strategy shows bad performance with large β_G.

D. Scalability Experiment

Results in the validation experiment considers limited setups about number of targets (m) and swarm size (n). In scalability experiment, we present the grid result of a series of m and n. The number of target varies from 10 to 100 and population from 10 to 50. Step size of both variables is 2.

Results are shown in Figure 4 in a four by two table. The first three columns shows the iteration results after collecting 50%, 75% and 100% targets and the last column is the total distance after collecting all the targets. The first row shows the result of IGES divided by that of GES and the second shows the division by RPSO. X and Y axis of each sub-figure stands for the population and number of targets and the color at every position represents value. The color map is shown on the right of the entire figure. Colder color indicates lower value and more advantage of IGES.

From the color map, the values in the figure ranges from about 35% to 90% which indicates that IGES is always better than other two algorithms. In fact, larger values only appears in the second row when the population is lower than 20. The cooperation of IGES and RPSO may have the similar effect in performance when population is not very large and target is very crowded in the environment. However, when the population grows, the difference is very obvious.

From the table, all three algorithms shows quite the same trend as the population and number of targets increase, as the sub-figures on left three columns shares the same shape except the colors changes. On the first row, color becomes warm and the advantage of IGES shrinks a little. On the second row, the trend of the color is the opposite. This is easy to understand, as strategy of RPSO shows better performance at the beginning of the simulation and GES searches quicker near the end, if only comparing the two algorithms.

Overall, IGES shows great advantage in iteration. It only takes 50-60% and 45-70% of iterations than RPSO and GES. Advantage of GES varies in a large range, since performance of GES is quite poor when either many targets remain in the environment or population exceeds certain limit. IGES proposes strategy focusing on such issues and the results shows the improvement is effective.

The trend of distance is quite stable when population and number of targets change. The three algorithm shares the same trend as distance is almost linearly related to the iteration from the results in previous experiment.

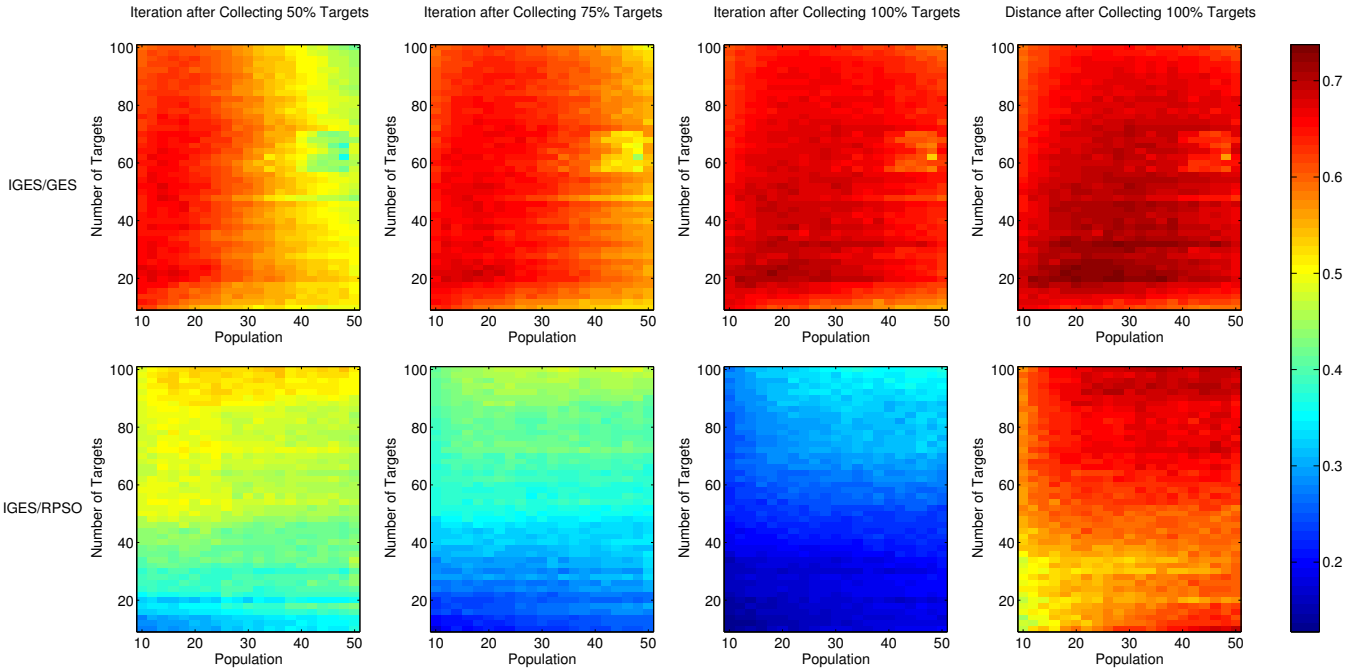


Fig. 4: Iteration and distance ratio results in scalability experiments.

V. Conclusion

An improved group explosion strategy (IGES) for searching multiple targets is proposed in this paper. We analyzed several shortcomings of the GES, proposed in our previous work, and improve the strategy focusing on these shortcomings. IGES is simulated in our self-built computer platform and compared with other two algorithms. Several experiments are conducted to test the algorithm in various aspects. The simulation results demonstrate that IGES shows great efficiency and stability in searching multiple targets regardless the number of targets or robots. IGES has only one parameter and very little computation in each iteration. IGES can adapt to different scales of targets and robots. The results indicate the cooperation in IGES can take advantage of small or large populations.

As for future work, we plan to focus on two aspects. The first aspect is to improve the strategy by introducing inter-group cooperation which is not very focused in both GES and IGES. Cooperation among groups can accelerate the searching progress. On the other hand, we will also try to apply IGES on more complicated searching problems with more restrictions, such as dynamic environment or various kinds of targets with different values.

References

[1] Q. Tang and P. Eberhard, "Cooperative motion of swarm mobile robots based on particle swarm optimization and multibody system dynamics," *Mechanics based design of structures and machines*, vol. 39, no. 2, pp. 179–193, 2011.

[2] J.-H. Lee, C. W. Ahn, and J. An, "A honey bee swarm-inspired cooperation algorithm for foraging swarm robots: An empirical analysis," in *Advanced Intelligent Mechatronics (AIM), 2013 IEEE/ASME International Conference on*. IEEE, 2013, pp. 489–493.

[3] M. Masár, "A biologically inspired swarm robot coordination algorithm for exploration and surveillance," in *Intelligent Engineering Systems (INES), 2013 IEEE 17th International Conference on*. IEEE, 2013, pp. 271–275.

[4] K. Derr and M. Manic, "Multi-robot, multi-target particle swarm optimization search in noisy wireless environments," in *Human System Interactions, 2009. HSI'09. 2nd Conference on*. IEEE, 2009, pp. 81–86.

[5] G. Cabrita and L. Marques, "Divergence-based odor source declaration," in *Control Conference (ASCC), 2013 9th Asian*. IEEE, 2013, pp. 1–6.

[6] H. Espitia and J. Sofrony, "Path planning of mobile robots using potential fields and swarms of brownian particles," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*. IEEE, 2011, pp. 123–129.

[7] D. Gong, C. Qi, Y. Zhang, and M. Li, "Modified particle swarm optimization for odor source localization of multi-robot," in *Evolutionary Computation (CEC), 2011 IEEE Congress on*. IEEE, 2011, pp. 130–136.

[8] W. Li and W. Shen, "Swarm behavior control of mobile multi-robots with wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1398–1407, 2011.

[9] Z. Zheng and Y. Tan, "Group explosion strategy for searching multiple targets using swarm robotic," in *IEEE Congress on Evolutionary Computation*, 2013, pp. 821–828.

[10] M. Couceiro, R. Rocha, and N. Ferreira, "A novel multi-robot exploration approach based on particle swarm optimization algorithms," in *Safety, Security, and Rescue Robotics (SSRR), 2011 IEEE International Symposium on*, nov. 2011, pp. 327 –332.

[11] X. Songdong, Z. Yunlong, Z. Jianchao, X. Zhibin, and D. Jing, "Group decision making aided pso-type swarm robotic search," in *Computer, Consumer and Control (IS3C), 2012 International Symposium on*. IEEE, 2012, pp. 785–788.

[12] J. McLurkin, A. J. Lynch, S. Rixner, T. W. Barr, A. Chou, K. Foster, and S. Bilstein, "A low-cost multi-robot system for research, teaching, and outreach," in *Distributed Autonomous Robotic Systems*. Springer Berlin Heidelberg, 2013, pp. 597–609.

[13] Z. Zheng and Y. Tan, "An indexed k-d tree for neighborhood generation in swarm robotics simulation," in *Advances in Swarm Intelligence*, ser. Lecture Notes in Computer Science, Y. Tan, Y. Shi, and H. Mo, Eds. Springer Berlin Heidelberg, 2013, vol. 7929, pp. 53–62.