# Distributed Multi-Robot Search in The Real-World Using Modified Particle Swarm Optimization

Amirali Darvishzadeh and Bir Bhanu
Center of Research in Intelligent System
University of California, Riverside
{darvisha@cs,bhanu@cris}.ucr.edu

## ABSTRACT

A challenging issue in multi-robot system is to design effective algorithms which enable robots to collaborate with one another in order to search and find objects of interest. Unlike most of the research on PSO (particle swarm optimization) that adopts the method to a virtual multi-agent system, in this paper, we present a framework to use a modified PSO (MPSO) algorithm in a multi-robot system for search task in real-world environments. We modify the algorithm to optimize the total path traveled by robots. Experiments with multiple robots are provided.

## Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems

## General Terms

Algorithms

## Keywords

multi-robot search, swarm intelligence

## 1. INTRODUCTION

Utilizing PSO method in a distributed multi-robot system potentially has two important problems: *First*, the method may guide the robots navigate through the regions that are already explored. *Second*, in some cases, although a robot may be close enough to find the target by performing a local search, but PSO guides the robot to locations which are farthest to the target from the current position of the robot. In other words, the robot does *not* search enough in promising search regions. To overcome the limitation of PSO when applied from virtual environment to real-world robotic system, we develop a framework called MPSO for a multi-robot system. We use a one-to-one match between particles in the PSO method and robots in a multi-robot search. We utilize a local search method to optimize the search time.

## 2. TECHNICAL APPROACH

Figure 1 shows an overview of our swarm intelligent system. During a search experiment, robot's camera captures images from the search environment. The images are sent to the object detection system to detect candidate targets. The biggest candidate target is selected as the target and based on its area in the image, fitness value is calculated.

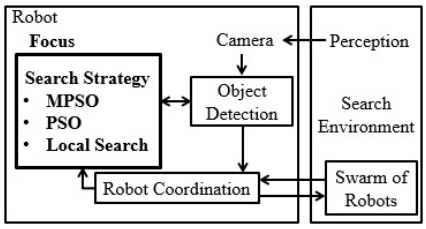



**Figure 1: System Framework**

• **Particle Swarm Optimization:** In the popular PSO method [3] new position to be explored by an agent is calculated as follows:

$$v_{i+1} = \omega v_i + \phi_p r_p(p_i - x_i) + \phi_g r_g(g - x_i) \quad (1)$$

$$x_{i+1} = x_i + v_{i+1} \quad (2)$$

where $x_i$ and $v_i$ are the current position and velocity of the agent at $i^{th}$ iteration of the algorithm, respectively. $x_{i+1}$ denotes the next position to be explored by an agent. Best solution found by particle and swarm systems are shown by $p_i$ and $g$, respectively. In addition, $r_p$ and $r_g$ are uniformly distributed random numbers from $(0, 1)$ range. $\omega$, $\phi_p$ and $\phi_g$ are acceleration constants and must be studied for the system to perturb $v_i$, $(p_i - x_i)$ and $(g - x_i)$ velocity vectors.

• **Modified PSO for Multi-Robot Search:** To overcome the problem of the PSO as outlined in section 1, we discretize the continuous search space in the map into non-overlapping similar squares that are called *units*, then we add the following criteria to the search strategy. (A) let for the next iteration of the PSO, a robot plans a path that contains $PATH_{units}$, the number of units along the path. Let $PATH_{visited}$ be the number of visited units and $PATH_{unvisited}$ be the number of unvisited units on the path. If $PATH_{visited} > PATH_{unvisited}$, then the robot will search in $PATH_{unvisited}$ units in its neighborhood instead of working with the PSO method. This would help the robots to explore more search regions in a shorter period of time. (B) at any time, if fitness value of a robot is greater than a certain amount of $\delta$, the robot will select a set of points of interest in its field of view to look for the target instead of working with the PSO algorithm. If fitness goes below $\delta$ the robot continues to look for the target by using PSO Equation ( 1) and ( 2).

• **Fitness Value:** In this work, fitness value is calculated as $Fitness = \frac{\alpha}{\beta}$ where $\alpha$ is the area of detected target and $\beta$ is the area of expected goal.


*GECCO'14,* July 12-16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07.
http://dx.doi.org/10.1145/2598394.2598405.

## 3. EXPERIMENTAL RESULTS

We have utilized a multi-robot system (P3AT, PeopleBot, PatrolBot made by Mobilerobot Inc.) and a mobile robot simulator (when swarm size is greater than three robots) to test the performance of the proposed algorithm in different settings. In our experiments, the goal is to find a target which is a colored object. For each test case, we have repeated the experiments for 10 times and plotted the average times. The values of PSO coefficients and inertia weigths are selected as in [2] ($\omega = \frac{1}{2\times ln(2)}$, $\phi_g = \phi_p = 0.5 + ln(2)$).

• **Scalability and Experiments:** We carried out several experiments to test the performance of MPSO vs. PSO, PSO-2S [1] and Exhaustive methods.

• **Evolution of parameter $\delta$:** To study the effect of $\delta$ parameter in MPSO, we run experiments with swarm population of 20 agents in the map of size $5000m^2$. We start experiments with $\delta = .1$, and in each step we add .1 value to it. The results are shown in Figure 2.

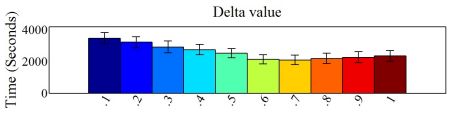
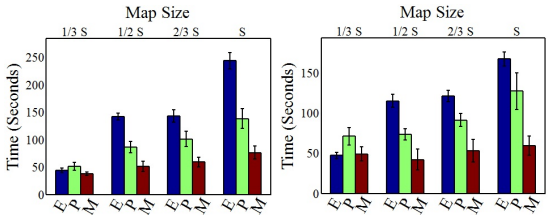



**Figure 2: E1. Evolution of parameter $\delta$**

Although the values in $[0.6, 1]$ range are close to each other, but the best results are obtained when $\delta = .7$.

• **Growing size of the search space:** We divide the search space ($S$) into different regions and in each test case we select a subset of $1/3S$, $1/2S$, $2/3S$ and $S$ to test our multi-robot system. For each number of robots (either two or three) utilized in the search task we repeat the experiment for ten times. The average of the times in the search tasks are shown in Figure 3.



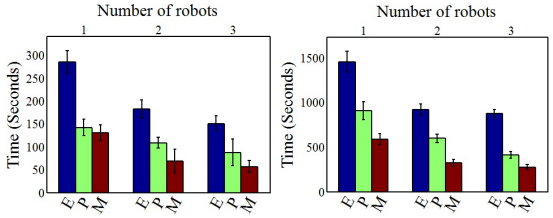

(a) Two-Robot Search (b) Three-Robot Search

**Figure 3: Performance of the methods in growing search space. Exhaustive-E, MPSO-M, PSO-P.**

• **Performance of the methods in Multi-Robot Search:** We conduct the experiments by utilizing one, two and three robots in two different search spaces ($S_1 = 142m^2$ and $S_2 = 946m^2$). Figure 4 shows that utilizing more robots in search task significantly improves the performance.

• **MPSO vs. PSO-2S [1]:** By adding more agents to the swarm system, obstacle avoidance and path replanning of the agents would be time consuming tasks which affects the performance of the system. Partitioning the search space and assigining a small number of agents to each partition would increase the performance of the system. We have used the method introduced in [1] to partition the search space into multiple zones and assign agents to the zones, and then compared our MPSO method with PSO-2S [1].

In Figure 5 and 6, on the average, MPSO is 13% and 21% faster than PSO-2S, respectively.




(a) Small Search Space (b) Large Search Space

**Figure 4: Performance of the methods in Multi-Robot Search. Exhaustive-E, PSO-P, MPSO-M.**

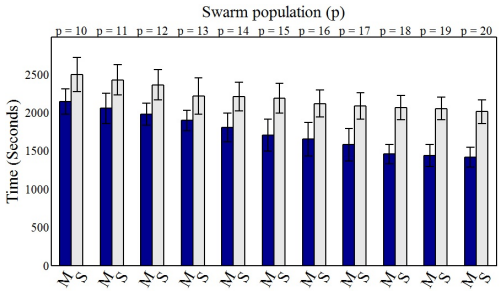



**Figure 5: Scalability of MPSO vs. PSO-2S. MPSO-M, PSO-2S-S.**

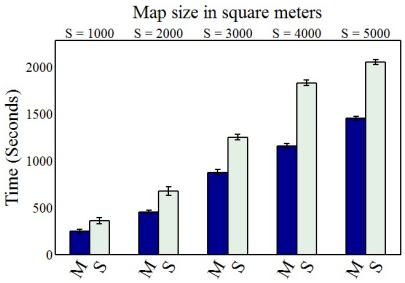



**Figure 6: MPSO vs. PSO-2S for different map sizes. MPSO-M, PSO-2S-S.**

## 4. CONCLUSIONS

We presented a modified PSO (MPSO) algorithm where two local search constraints are added to PSO in order to enhance its performance in multi-robot search task in the real-world. Results show that MPSO outperforms PSO and PSO outperforms the exhaustive search.

## 5. ACKNOWLEDGMENT

This work was partially supported by NSF grant 0727129.

## 6. REFERENCES

[1] A. El Dor, C. Maurice, and S. Patrick. A multi-swarm PSO using charged particles in a partitioned search space for continuous optimization. *Computational Optimization and Applications*, 53(1):271–295, 2012.

[2] S.-T. Hsieh, T.-Y. Sun, C.-C. Liu, and S.-J. Tsai. Efficient population utilization strategy for particle swarm optimizer. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(2):444–456, 2009.

[3] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proc. 1995 IEEE Int. Conf. Neural Networks IV*, pages 1942 –1948 vol.4, Nov/Dec 1995.