NUWC-NPT Technical Report 12,318
6 June 2019

Control of Multi-Agent Swarms with Cooperative Particle Swarm Optimization

Kyle D. DeMedeiros
Undersea Warfare, Weapons, Vehicles, and Defensive Systems Department

Thomas A. Wettergren
Code 00T1 — Senior Technologist (NUWC Division Newport)

NEWPORT

Naval Undersea Warfare Center Division Newport, Rhode Island

PREFACE

This report was funded under NUWC Division Newport Network Activity No. 100001450919 0010, “Swarm Dynamics/Performance,” principal investigator Kyle D. DeMedeiros (Code 8512). The sponsoring activity is the Office of Naval Research: Sea Warfare and Weapons Department (ONR Code 333WP).

The technical reviewer for this report was David B. Segala (Code 60E).

Reviewed and Approved: 11 September 2019

Hector M. Lopez
Head, Undersea Warfare, Weapons, Vehicles, and Defensive Systems Department

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From – To)
06-06-2019	Technical Report	

4. TITLE AND SUBTITLE
Control of Multi-Agent Swarms with Cooperative Particle Swarm Optimization

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Kyle D. DeMedeiros
Undersea Warfare, Weapons, Vehicles, and Defensive Systems Department

Thomas A. Wettergren
Code 00T1 — Senior Technologist (NUWC Division Newport)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Undersea Warfare Center Division
1176 Howell Street
Newport, RI 02841-1708

8. PERFORMING ORGANIZATION REPORT NUMBER
TR 12,318

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Office of Naval Research
875 N. Randolph St,
Arlington, VA 22217

10. SPONSORING/MONITOR'S ACRONYM
ONR

11. SPONSORING/MONITORING REPORT NUMBER

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report describes a new approach for the distributed control of multi-agent systems that are performing search in uncertain environments. This approach is called Cooperative Particle Swarm Optimization (Cooperative PSO), and is derived as a modification of Particle Swarm Optimization (PSO) that accounts for uncertainty in the search environment. This research develops a Cooperative PSO technique and illustrates how it can be applied to physical search systems (such as robotic swarms) to successfully control the cooperative behavior of a small swarm of agents. Simulation experiments are performed to show the algorithm's effectiveness via comparisons against conventional PSO in scenarios with uncertain search environments.

15. SUBJECT TERMS
particle swarm optimization, multi-agent search, autonomous agents, swarms, metaheuristics, cooperative control

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
(U)	(U)	(U)	SAR	35	Kyle D. DeMedeiros

19b. TELEPHONE NUMBER (Include area code)
401-832-5831

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

AOI Area of Interest
MAS Multi-Agent Search
PSO Particle Swarm Optimization
SPSO Standard Particle Swarm Optimization

CONTROL OF MULTI-AGENT SWARMS WITH COOPERATIVE PARTICLE SWARM OPTIMIZATION

1. INTRODUCTION

Multi-agent search (MAS) is a process consisting of the cooperation between multiple entities, known as agents, in the analysis of information from an environment to find a meaningful solution to a problem. These MAS problems can arise in physical search settings (such as finding hidden objects) or in informational settings (such as finding a piece of information hidden in data). In either application setting, the agents in an MAS system are performing individually to cooperatively reach the goal. There are multiple ways to perform MAS, including:

- Sharing found information to reduce the search space amount that each individual agent must cover (search area reduction),
- Utilizing multiple agents to repeatedly search the space to avoid missing small regions due to agent error (search overlap), and
- Avoiding false alarms/false positives in the search (redundant search),

Search can also be static (pattern search), or developed so that each agent is used to its full potential (optimal search). A popular avenue of MAS currently includes swarm techniques. In this context, a swarm is a group of simple agents interacting with the environment and each other to create complex behaviors[1].

Swarm techniques emulate natural processes, which leverages generations of naturally tested techniques. These techniques are attractive for MAS because they have the benefit of being evaluated by the organisms using them in environments of interest. Nature-inspired techniques also show redundancy of use (for instance, swarms of insects on land, birds in the air, and fish under the sea), which further illustrates their usefulness. In addition, swarm techniques generally utilize aggregates of simple rules to generate emergent behavior. This emergent behavior generates not-quite-optimal, but high-performance searches. These simple rules are also easy to implement in software, quick to run on standard computer processors, and easy to modify based on specific problem requirements. In the literature, this emulation shows up in many forms ranging from mimicry of animal and insect behaviors[2], to human-like interactions and decision processes[3].

When looking at swarms in this context, their capabilities as an MAS for problem solving become apparent. Swarm techniques focus almost entirely on distributed control, whereas other types of MAS either utilize a centralized controller[4, 5], or very rigid formation-based distributed control[6, 7, 8]. This focus on decentralized control grants swarms benefits that other techniques (such as those that rely on specific groupings, patterns, and formations of agents) do not, such as:

- Any agent can perform any task (redundancy),
- Group formation can become irrelevant (formation independence), and
- If some agents fail, the group can continue the mission (group survivability).

Due to the popularity of swarm techniques, there are a large number of different techniques available that are geared to a large variety of problems and inspired by a myriad of natural processes. Techniques such as genetic algorithms[9], ant colony optimization[10], simulated annealing[11], particle swarm optimization[12], artificial bee colony[13], glowworm swarm optimization[14], and gravitational search algorithm[15] all utilize natural swarm and swarm-like processes to find solutions to various problems. Of particular interest recently is particle swarm optimization (PSO). Section 2 covers the general formulation of PSO and its use.

All MAS techniques rely on information gathered about the search environment. This information is graded (using an objective function) and acted upon based on the technique used. This grading often relies on information density (most likely location, least likely location, highest probability of detection/success, etc.). But, the techniques using these functions don't take into consideration how uncertain the information being sensed may be (i.e., large object likelihoods may be due to a static object radiating information, but changes when the object moves; and large detection probabilities are often due to a homogeneous static environment and decrease dramatically when that changes). The previously mentioned swarm techniques are all successful MAS techniques; however, these techniques do not focus on this uncertainty of the information space being worked.

In this technical report, researchers used a recently-developed algorithm derived from PSO called cooperative particle swarm optimization (Cooperative PSO). This algorithm marries the benefits of swarm techniques (in this case, Standard PSO (SPSO)) with the robustness of group cooperation to explore an uncertain environment[16]. Cooperative PSO is a swarm search algorithm that uses group and individual information to adjust each agent's movement about the information space. In this way, Cooperative PSO is similar to SPSO; however, Cooperative PSO differs from SPSO in how information from other agents is used and combined. Cooperative PSO protects against local optima by not only finding the "best" locations in the space already searched, but by searching around this "best" location in order to gain more knowledge about the environment itself. This focus on both exploration and exploitation is the key difference between Cooperative PSO and other swarm techniques.

This technical report is organized as follows. Section 2 provides an overview of SPSO in the way it is applied herein. Section 3 describes the Cooperative PSO algorithm and provides a discussion of how it functions. Section 4 outlines the computational experiments performed and the metrics used to evaluate Cooperative PSO. Section 5 describes the numerical results seen in the experimentation. The report concludes with a discussion of lessons learned and future work.

2. PARTICLE SWARM OPTIMIZATION

PSO was developed by Kennedy and Eberhart in 1995 to mimic bird flocking patterns when finding sources of food[12]. Since then, PSO has become a popular swarm technique for a myriad of problems including multi-objective optimization[17, 18, 19, 20], optimal design[21, 22], multi-swarm control[23, 24, 25], and robot swarm control[26].

In the standard implementations of PSO, there are three mechanisms governing how swarm agents move to new search locations: (1) inertia, (2) personal history, and (3) group history. The inertia mechanism is for movement based on the agent's previous movement in the search space. The personal history mechanism is for movement based on the best-known information that agent has observed up to present time in the search, and the group history mechanism is for movement based on the best-known information that the group as a whole has observed up to present time in the search. These three mechanisms allow the individual agents within the swarm to move about the search space to find a near-optimal solution to a large variety of problems. However, reference 27 relates an in-depth analysis of PSO and its various derivations, showing PSO limitations as well as the different methods used to supplement, mitigate, or eradicate these limitations. Another of these limitations, and the focus of this work, is uncertainty in the search space and how it can be reduced.

PSO can be applied as a multi-agent motion planning algorithm for physical search agents (such as a group of robots) by allowing the PSO mechanisms to provide dynamic adjustment of each individual agent's velocity vector. The adjustment of the velocity vector in such a physical search approach is based on an additive weighting of three contributing biases:

1. The agent's inertia tending to follow its current velocity,
2. A velocity bias to move the agent in the direction of that agent's "best" past position, and
3. A velocity bias to move the agent in a direction given by available history of all other agents (group consensus). This bias element is based on following the "best" past position of any of the agents, as that will draw all agents towards a common goal.

While there are many variants of PSO, this research uses SPSO[27] as an example to compare with Cooperative PSO. PSO follows a general framework for progressing the agents within the swarm by determining an agent's velocity as a weighted combination of three components as follows:

$$v_{i+1}^k = u_1 I\left(v_i^k\right) + u_2 P\left(x_i^k, x_{best}^k\right) + u_3 G\left(\left\{x_i^j\right\}_{j=1}^K, \left\{x_{best}^j\right\}_{j=1}^K\right) \tag{1}$$

where $I(v_i^k)$ is a term representing the agent's inertia, $P(x_i^k, x_{best}^k)$ is a bias to draw the agent towards its best past progress, and $G(\{x_i^j\}_{j=1}^K, \{x_{best}^j\}_{j=1}^K)$ is a bias to draw the agent towards some combination of the set of agents (including their current locations and/or their best locations). The weightings $\{u_1, u_2, u_3\}$ are algorithm design parameters that provide a preference between these three components of motion planning. This research notes that the inertia weighting, $u_1 = c_I$, is chosen as a fixed deterministic value while the other weights, $\{u_2, u_3\}$, are stochastic terms that vary over time steps to provide a random switching between biases towards individual and group performance. This research specifically uses $u_2 = c_P\, \xi_P$ and $u_3 = c_G\, \xi_G$ with constant weights $\{c_P, c_G\}$ and random components $\xi_P \in U\,(0,1), \xi_G \in U\,(0,1)$ that are uniformly random over the unit interval. This type of stochastic switch has been shown to improve the search heuristic.

For the SPSO version of swarm optimization, the inertia term is simply given by

$$I\left(v_i^k\right) = v_i^k \tag{2}$$

and allows agent k to have a preference to continue moving at the same velocity (corresponding to physical inertia). The personal bias term $P(x_i^k, x\,_{best}^k)$ is a bias to create a "pull" of agent k towards whichever location created the "best" performance for the individual agent (denoted as x_{best}^k); note that this view of best is typically taken to be the location where the greatest objective achievement had been obtained. In particular, for a swarm optimization objective $J(x)$ that is a function of individual agent position, the best performance for agent k at time step i is given by

$$x_{best}^k = \arg\max_{\{x_\ell^k\}} J\left(x_\ell^k\right), \qquad \ell = 1, \ldots, i. \tag{3}$$

Thus, the personal best term in SPSO is written as

$$P\left(x_i^k, x_{best}^k\right) = \frac{x_{best}^k - x_i^k}{\delta t}, \tag{4}$$

where δt is the time step interval (step from i to $i+1$). The group bias term $G\left(\left\{x_i^j\right\}_{j=1}^K, \left\{x_{best}^j\right\}_{j=1}^K\right)$ is similar to the personal best bias in that it is chosen to provide a draw towards a location where the agent has had success. However, in contrast to the personal best, the group best is drawn towards the best of all agents' past performance, g_{best}, given by the x_{best}^k location that provided the greatest objective achievement over all k's. Therefore, the group bias term in SPSO is written as

$$G\left(\left\{x_j^i\right\}_{j=1}^K, \left\{x_{best}^j\right\}_{j=1}^K\right) = \frac{g_{best} - x_i^k}{\delta t}, \tag{5}$$

where g_{best} is the best location from any of the agents over all previous time steps, specifically given by

$$G_{best} = \arg\max_{\left\{x_{best}^j\right\}} \left(J_{best}^j\right), \qquad j = 1, \ldots, K. \tag{6}$$

Thus, for SPSO, the swarm equation of equation (1) takes on the following form:

$$v_{i+1}^k = c_I v_i^k + \left(\frac{c_P\, \xi_P}{\delta t}\right)\left(x_{best}^k - x_i^k\right) + \left(\frac{c_G\, \xi_G}{\delta t}\right)\left(g_{best} - x_i^k\right), \tag{7}$$

which is the standard PSO form used in the literature.

3. COOPERATIVE PARTICLE SWARM OPTIMIZATION

Cooperative Particle Swarm Optimization (Cooperative PSO) is a swarm-based metaheuristic inspired by Particle Swarm Optimization (PSO)[16]. It is inspired in the manner of using the coordination of agents (particles) as in PSO by applying biases to each agent's current velocity vector that are based on information gained from both the past history of the individual agent as well as the available history of other agents.

Cooperative PSO differs from PSO in that it focuses on the reduction of information uncertainty, or more specifically, to increase the overall knowledge of an area of interest (AOI). This cooperative reduction of information uncertainty allows Cooperative PSO to better integrate exploration of the environment with exploitation of the local optima, reducing premature convergence to a local optima. In reference16, the concept of Cooperative PSO (then called CSO) is initiated through some heuristic arguments and examples are provided on guiding a small number of agents effectively.

This technical report more thoroughly derives the Cooperative PSO algorithm and shows some parametric variations to assess its performance as a multi-agent motion control algorithm across a range of simple scenarios. Also, this research further extends the initial Cooperative PSO concept by introducing some dynamic variation in previously fixed parameter values.

3.1 UPDATING UNCERTAINTY WITH AGENT OBSERVATIONS IN COOPERATIVE PSO

Before describing the structure of Cooperative PSO, how it utilizes the information from the swarm to achieve the desired goal must be described. To accomplish this cooperative goal, each agent traverses the information space by considering the agent's current information (the information it can sense/record right now), the agent's personal history (where it has been and the relevance of the information at that point), and the group's information (what each group member sees and where they are located within the information space). This information "pulls" the agent in space, and each of these driving forces are weighted differently to manipulate each component's importance on the agent's velocity.

To track how each agent searches the information space, the researchers used data determined from the information space and based on the reports of agents that have previously searched the information space. This information is referred to as the uncertainty value. In an effort to represent potentially limited communications bandwidth between agents, each agent only sends their position (parameter data) and their uncertainty value to agents in the swarm. The relative importance of this information to an individual agent's motion plan is reduced by the distance the sender is from the receiving agent. In this way, Cooperative PSO can operate in communications-restricted environments, such as low-powered computers, in robotics with limited communications bandwidth, in systems with high corruption of transmitted data, etc.

The procedure by which the algorithm uses agent-collected information to inform others is based on the notion of a reduction of uncertainty. Let $\eta_i^k(x)$ be agent k's understanding of the phenomenon of concern at location x at time step i. Furthermore, let $\alpha_i^k(x)$ be agent k's observation/measurement of the phenomenon at location x at time step i. Uncertainty of x is reduced when agent k visits the area. The rate of reduction is directly related to whether the agent's prior understanding is in agreement with $\alpha_i^k(x)$ (faster reduction), or is in conflict (slower reduction). By sharing in this uncertainty measure, the Cooperative PSO algorithm encourages the agents to head to areas where there is more uncertainty (thus increasing the exploration of the space). Furthermore, complete removal of uncertainty throughout the workspace will lead to finding any objects that are sought or a complete measurement of the phenomenon of interest, depending upon the goal of the group operation. The shared group understanding of the uncertainty with which the phenomenon can be observed is $\lambda(x)$. Note that agents do not share the measurements of the region, nor the agent's individual understanding of the phenomenon, only the uncertainty.

The researchers next defined update rules for η and λ based on observations α. The update for the expected measurement value of the individual agents is defined according to the following update rule:

$$\eta_i^k(x) = \lambda_{i-1}(x) \cdot \alpha_i^k(x) + \left(1 - \lambda_{i-1}(x)\right) \cdot \eta_{i-1}^k(x) \tag{8}$$

This update uses the location uncertainty $\lambda_{i-1}(x)$ as a relative weighting between the measurements $\alpha_i^k(x)$ and the agent's prior understanding $\eta_{i-1}^k(x)$. To initialize the system process, all of the values of $\eta_k^i(x)$ for all of the agents are set to a known (known to all the agents) nominal level $\eta_0(x) = \eta_i^k(x)$ at $i = 0$ for all k. This is the assumed understanding of the phenomenon *a priori*. The only requirement on the values of $\eta_0(x)$ is that $\eta_0(x) > 0$.

Furthermore, the location uncertainties are initialized to begin at the maximum uncertainty, such that $\lambda_0(x) = 1$ for all x. Note that this updating rule is a simple weighting of the measured value of the phenomenon and the agent's understanding of the phenomenon at that location. The weighting values have been chosen such that the weighting biases strongly towards the new measurements when there is high uncertainty (low confidence) in the prior understandings (as $\lambda_{i-1}(x) \to 1$), and conversely the weighting biases strongly towards the prior understandings when there is low uncertainty (high confidence) in those understandings (as $\lambda_{i-1}(x) \to 0$).

The value of uncertainty $\lambda_i(x)$ is updated according to the information collected. To do this, it is necessary to first compute the observed measurement discrepancy $\psi_i^k(x)$, which measures the difference between what has been measured of the phenomenon and what the agent previously understood of the phenomenon. This is given by

$$\psi_i^k(x) = \left|\frac{\alpha_i^k(x) - \eta_{i-1}^k(x)}{\eta_0(x)}\right|. \tag{9}$$

A large value of $\psi_i^k(x)$ corresponds to a measurement that is vastly different than the prior understanding, whereas a small value corresponds to a measurement that meets the current understanding. This discrepancy value is used to update $\lambda_i(x)$ through the following update rule:

$$\lambda_i(x) = \lambda_{i-1}(x)\left[1 - \exp\left(-\psi_i^k(x)\right)\right]. \tag{10}$$

Note that $\lambda_i(x)$ is always decreasing in a static environment, as there is always benefit in searching an area. The frequency of searches to a particular area will determine how rapidly $\lambda_i(x)$ decreases.

If agents are deployed in an environment where it is expected that the underlying measurement quantities are dynamic (such as rapidly changing temperatures in an area, a moving object, etc.), then a simple fading memory can be applied to the shared values of $\lambda_i(x)$ via

$$\lambda_i(x) \longmapsto \gamma + (1-\gamma)\lambda_i(x) \tag{11}$$

for some value of $0 \leq \gamma < 1$ (where $\gamma = 0$ corresponds to no fading). The fading memory has the effect of increasing all of the uncertainty values in a proportional manner. This fading memory can be applied intermittently or periodically as desired to account for the expected dynamics. This can be achieved by setting the value of fading memory based on the dynamicity level of the environment.

3.2 STRUCTURE OF THE COOPERATIVE PSO

For Cooperative PSO, while the general form of swarm as shown in equation (1) is still used, the manner in which the group component acts on the information gathered is much different. Cooperative PSO uses the information content in a swarm-based algorithm that is similar in structure to PSO, with a main difference being in how the third bias element is calculated. This third bias is based on having the agent following a weighted average of current positions of all other agents. This still keeps all agents near one another, but also maintains effectiveness in exploration of the AOI. Cooperative PSO still uses the same form for the inertia term as in equation (2). In reference 16, the personal best bias $P(x_i^k, x_{best}^k)$ that was initially developed for Cooperative PSO was given with a weighting of $c_P = \lambda_{best}^k$.

This type of dynamic weighting contrasts with the static weighting used in SPSO, as it decreases the relative importance of the individual bias the more uncertainty is reduced. This dynamic weighting has an effect of causing the swarm to become driven more by group biases as the operation progresses and uncertainty is reduced. Also, it can pull the swarm back towards more individual biases if an event occurs to increase uncertainty (or the group's uncertainty reduces faster than the individual agent's uncertainty), thus providing a robustness to changes in the environment. This research examines the implications of this dynamic weighting as part of these simulation experiments. As shown in section 4, to create as close a comparison as possible (for part of the experimentation), the Cooperative PSO's personal component was set to be identical to SPSO's. The Cooperative PSO and SPSO was set to use the traditional form of Cooperative PSO's personal component as shown in equation (4). In this way, it can be determined how the individual changes from Cooperative PSO to SPSO affect the results.

The group bias term in Cooperative PSO is also weighted by the uncertainty reduction that has been seen. However, rather than utilizing the group best g_{best} as in SPSO, the Cooperative PSO approach uses a weighted distance between the given agent and each of the other agents. In particular, it weights these distances by using the individual agent's current uncertainty reduction as the weight. In this way, agents that experience higher uncertainty will tend to draw the swarm of other agents toward them, and additionally the group effect of a cluster of many agents with limited uncertainty reduction will be reduced. The specific form for the group bias that is used by Cooperative PSO is thus given by

$$G\left(\left\{x_i^j\right\}_{j=1}^K, \left\{x_{best}^j\right\}_{j=1}^K\right) = \frac{1}{K}\sum_{j=1}^K \lambda\left(x_i^j\right)\left(\frac{x_i^j - x_i^k}{\delta t}\right), \tag{12}$$

such that the swarm equation of equation (1) takes on the following form:

$$v_{i+1}^k = c_I v_i^k + \left(\frac{c_P \xi_P}{\delta t}\right)\left(x_{best}^k - x_i^k\right) + \left(\frac{c_G \xi_G}{K\delta t}\right)\sum_{j-1}^K \lambda\left(x_i^j\right)\left(x_i^j - x_i^k\right). \tag{13}$$

Each agent must keep track of the most relevant information seen by the agent so far. This is their λ_{best}^k and x_{best}^k. These are determined based on the largest discrepancies seen by an agent between $\alpha\,_i^k(x)$ and $\eta\,_{i-1}^k(x)$ (i.e. a large value of $\psi\,_i^k(x)$). These are given by

$$x_{best}^k = \left\{x_{i^*}^k \,:\, i^* = ar\ gmax_i\left(\psi_i^k(x)\right)\right\} \tag{14}$$

and

$$\lambda_{best}^k = \lambda_{i^*}\left(x_{best}^k\right). \tag{15}$$

Thus, x_{best^k} is the location of λ_{best}^k and λ_{best}^k is the area of highest uncertainty seen by agent k thus far. Using the updated values of $\lambda_i(x)$, λ_{best}^k, and x_{best}^k, the Cooperative PSO algorithm provides motion plans for each agent in the group.

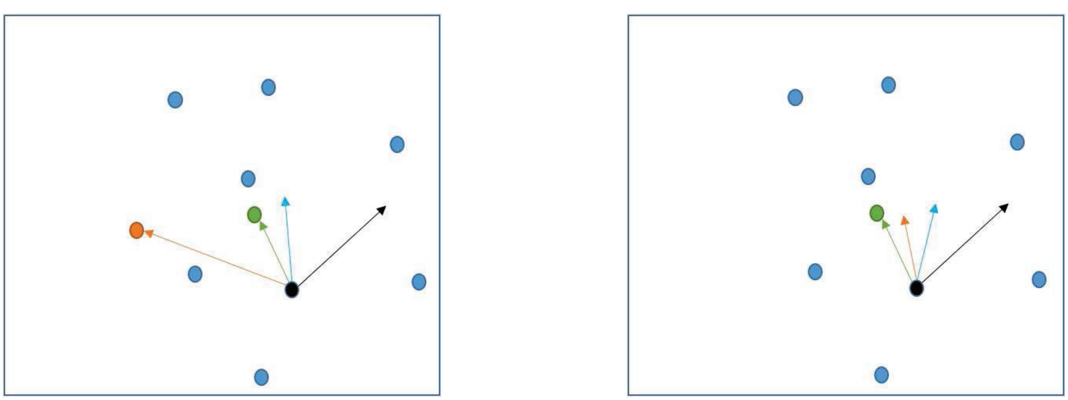
3.3 ALGORITHM DESCRIPTION

The individual agent velocity update equation as given in equation (13) is the main component of the Cooperative PSO algorithm. This algorithm comprises each agent's distributed motion planning controller and is the driving force for the agents moving throughout the group. The specific details of the Cooperative PSO algorithm implemented for each agent is given in Algorithm 1.

Algorithm 1 Cooperative PSO Algorithm

Read in a new value of $\alpha_i^k(x)$;
Compute $\psi_i^k(x)$ from $\alpha_i^k(x)$ and $\eta_{i-1}^k(x)$ using eq. (9);
Compute $\eta_i^k(x)$ from $\alpha_i^k(x)$, $\eta_{i-1}^k(x)$, and $\lambda_{i-1}(x)$ using eq. (8);
Compute $\lambda_i(x)$ from $\lambda_{i-1}(x)$ and $\psi_i^k(x)$ using eq. (10);
for $k = 1$ **to** K **do**
  if $\psi_i(x_{i-1}^k) \geq \psi_{best}^k$ **then**
    $x_{best}^k = x_{i-1}^k$
    $\lambda_{best}^k = \lambda_i(x^x)$
  end if
end for
for $p = 1$ **to** P **do**
  if $x_p \neq x_{i-1}$ **then**
    update λ_p via γ using equation (11)
  end if
end for
Update λ map and share $\lambda_i(x)$ with other agents;
Compute v_i^k using equation (13);
if $v_i^k > v_{max}$ **then**
  $v_i^k > v_{max}$; {for restricting agent speed}
end if
Set $x_i^k = x_{i-1}^k + v_i^k \delta t$

Figure 1 illustrates the schematically the difference between the Cooperative PSO and SPSO algorithms. In each algorithm, there is an inertial component (represented by the black line), a personal component (represented by the green line), and a group component (represented by the orange line). In both cases, the personal component is driven by the agent's personal best (represented in figure 1 by the green dot). In SPSO, this group component is driven by the group best (represented in figure 1a by the orange dot). In Cooperative PSO, this group component is a weighted sum of the group's current locations, and is not reliant on a "group best."



(a) Standard PSO ***(b) Cooperative PSO***

Figure 1. Standard PSO and Cooperative PSO Graphical Representation

After the calculation is complete, there is a resultant velocity (represented in figure 1 by the blue line). The main difference between SPSO and Cooperative PSO is this group component. SPSO's group best is an historical value that has a specified weight associated with it (this research used a static weight of $c_G = 1.5$). With Cooperative PSO, each component in the group (i.e. each agent in the swarm) has its contribution to the group further weighted by the agent's current λ value. The λ values can range from $0 \leq \lambda \leq 1$ and decrease over time as agents re-examine previously explored areas. This allows agents with newer information (higher λ values) to have more pulling power than agents with older information. In addition, since Cooperative PSO uses the current λ and position values of each agent, the group bias does not rely on a historical value, but a more dynamic one. Furthermore, since the λ values decrease with time, agents will have more tendency to be drawn to newer areas, rather than areas already searched.

The λ update procedure allows agents within the group to update spaces searched by other agents. This update causes other agents' personal bests to be worth less than they were originally as a result of the group re-searching areas already searched. This acts to "reset" an agent's best values to newer areas being searched. This update still protects against agents choosing an area of low λ as a new personal best and has the added effect of allowing an agent to choose new personal bests close to areas of high interest (due to areas of high interest requiring more search passes to lower their λ values). For both SPSO and Cooperative PSO, this acts like an anchor when the entire group is not finding any new information, allowing the group as a whole to converge closer to areas of higher interest. For SPSO, this also modifies the group best location and can act to dramatically pull agents toward areas of high interest (since both the agents' personal bests and the group best could be in the same area). This is beneficial for exploitation, but can hurt exploration. For Cooperative PSO, since there is not a group best, the agents are more likely to move slower to the areas of higher interest, allowing for more exploration (but requiring more time to exploit the AOI). The result is a much more balanced reliance on both the personal history and group components for the Cooperative PSO equation, creating an orbit-like movement of the agents and increasing exploration with little to no detrimental effect on exploitation.

Both the SPSO and Cooperative PSO algorithms perform planning by computing a potential change to agent velocity. It is noted that this change in velocity does not need to be computed on a regular scale. In particular, when communications issues prohibit transfer of information between agents, any agent will keep moving with its velocity as previously computed. When information required to update the velocity can be computed, the new velocity is determined from the appropriate swarm algorithm as any changes to velocity can be implemented on the vehicle. This allowance for intermittent communications makes these swarming algorithms very practical in many applications with unreliable communications links between the agents.

3.4 ALGORITHM CONSIDERATIONS

An in-depth review of PSO and many of its recent derivations (including what is referred to as SPSO) was performed in the research documented in reference 27. The information considered in that review is used to discuss Cooperative PSO in the context of other similar swarm-based metaheuristics. In particular, reference 27 contains multiple areas of interest used to categorize PSO and to evaluate the effectiveness of its iterations. Of these areas, five are relevant to Cooperative PSO: (1) network topology, (2) convergence to a point, (3) convergence to local optimum, (4) parameter setting, and (5) modification of the velocity update rules.

The network topology refers to the graphical interconnection between each agent in the swarm. By defining a topology for a given swarm algorithm, one can see how the connections between agents can affect the system's explorative and exploitative behavior (specifically by imposing different speed and propagation of information between agents)[27]. SPSO utilizes a global-best topology, whereas in Cooperative PSO, the concept of a global best is not used.

Instead, there is a reliance on the aggregate information seen by each agent at a particular time step i. The aggregate of this data pulls the group toward the inside of the group's perimeter. In this way, Cooperative PSO mitigates the potential of swarm optimization algorithms similar to it to fall into local optima (when the agents of a swarm converge to a point that is not a true optimal or near-true optimal).

Convergence to a point is a desired property of many swarm algorithms as it reduces the chance of swarm explosion. Swarm explosion is the increase of acceleration and inertial coefficients causing the velocity vectors of the particles to move toward infinity[27]. The earliest ways of combating this is by either (or both) restricting agent velocity to a bound, and restricting the agent position to a bound. The former does not stop swarm explosion, it only slows it. The latter will stop it, but at the expense of prematurely forcing convergence to a point of the algorithm (and potentially hindering the effectiveness of the algorithm). In this work, both Cooperative PSO and SPSO are restricted this way. Since both algorithms are restricted the same way, it is believed that the negative effect on the algorithm performance is irrelevant, since Cooperative PSO is compared to SPSO under the same conditions. In-depth analysis of how to bound the agents to their space was out of the scope of this work.

With respect to convergence to a point, the goal of Cooperative PSO is to increase exploration while still enabling exploitation of the system. As a result, Cooperative PSO protects against premature convergence to a point by implementing a biased weight of all agent locations. Since this bias is based on relative locations of agents, the bias is different for each agent and thus tends to avoid agents clumping together. This topology allows the agents more dynamic movement due to more dramatic changes in group bias as opposed to that found with SPSO's global best. This dynamic movement also increases exploration as the agents tend to "orbit" the group center since the group bias moves as each agent moves.

Cooperative PSO also seeks to avoid convergence to a local optimum by increasing exploration via reduction of uncertainty. The benefit is that Cooperative PSO can always be modified to reference a particular local optimum if no other better optimum was found. Keep in mind that another goal of Cooperative PSO is to increase exploration while exploiting the space as well. To do this, the λ calculations were made and used as replacements for the acceleration coefficients, and the group best was removed in favor of a weighted bias approach. Cooperative PSO also focuses on uncertainty. This means that once a space was checked, whether that space was optimal or not, searches still occur around that location to ensure it is indeed the optimal or near-optimal point. This means that subsequent visits to that "optimal" point will actually cause the λ values for that point to be reduced, effectively "releasing the anchor" to it. In situations where Cooperative PSO needs to maintain areas of interest, it can be modified to always be interested in particular areas. This, however, was out of the scope of this current work. During experimentation, the way λ is calculated effectively allows Cooperative PSO to reach the optimum or near-optimum point, rather than search about and away from it. Since λ is calculated using information from the environment, it can always record the points of interest for the user when the goal is detection of areas of interest rather than exploring those areas.

A final categorization for assessing the performance differences in variants of PSO is to examine how they modify the velocity and/or position update rules that govern the movement of agents within the system. Cooperative PSO would be a modification of the velocity rule within traditional PSO implementations (i.e., SPSO). This is accomplished by utilizing the λ values to replace the group best with the weighted sum from the agents in the swarm. The λ values are measurements of the system's entropy in that the λ values are the calculated results of previous information from the agent, information from the group, and information from the environment. The calculated value is then used as a weight to determine the direction the agent should move; as opposed to SPSO that uses static weights that are empirically determined, then evaluates the positions of the agents to determine the group bias for the update.

4. SIMULATION EXPERIMENTS

To illustrate the effectiveness of Cooperative PSO in group motion planning, the research simulates a group of distributed agents performing a two-dimensional search application. The agent motion is confined to a fixed area of interest (AOI) $\mathcal{W} \in \mathbb{R}^2$ and the agents all share an initial prior expectation of the object location, with that expectation represented as $\alpha(x) : \mathbb{R}^2 \rightarrow [0, 1]$ whose value corresponds to the prior likelihood that the object of search is at position x. The specific functional form of this prior $\alpha(x)$ is referred to as the "environment" of each simulation run. The weight of the inertial, personal, and group components of Cooperative PSO and SPSO (labeled c_I, c_P, and c_G, respectively) are adjusted to determine the relative performance of Cooperative PSO and SPSO in multi-agent search applications.

Four different instantiations of $\alpha(x)$ are considered to test performance on problems of varying complexity for the search. The first three environments consist of a simple Gaussian hump to represent a prior expectation of a likely location of the object of search. These cases vary in the specific location of the center of the hump as well as standard deviation of the Gaussian (varying as $\sigma = L/5, \sigma = L/2$ and $\sigma = L$ for the three cases, respectively, where L is the length of the AOI $\mathcal{W}$, which is assumed to be square in these examples). The fourth environment is similar to the previous three environments (with a small standard deviation on a Gaussian hump) but also includes multiple smaller amplitude humps to represent alternative likely object locations. In all cases, the object's "true" location is positioned near the center of the Gaussian hump.

In reference 27, a specific range is shown for the inertial coefficient and the acceleration coefficients that are seen as valid for most tests. The weighting for SPSO is as follows: $0.4 \leq c_I \leq 1.1$, $1.5 \leq c_P \leq 3$, and $1.5 \leq c_G \leq 3$. For this work, SPSO was set as $c_I = 1$ and $c_G = 1.5$. Cooperative PSO was also set as $c_I = 1$ and $c_G = 1.5$ for the group weights. Initial empirical testing found that these values worked well for this testing. And, since they fall within the expected ranges according to reference 27, it is believed that they are valid for these experiments. c_P is varied to explore the effect of personal history component on both SPSO and Cooperative PSO. Each of these experiments was run with 2000 time iterations and $n_{tot} = 1500$ Monte Carlo iterations, with each Monte Carlo iteration varying both the center of the environmental variation (the Gaussian hump) and the initial agent locations.

The goal of the search operation is to search as much of the space as possible while focusing on finding objects of interest. This can be seen as effectively searching as much of the region as possible for an object while limiting premature convergence. Thus, the following three metrics are considered to assess the performance of Cooperative PSO relative to SPSO: (1) success rate, (2) distance to the search object, and (3) amount of the AOI covered in search.

The success rate, $R(i)$, is given as the fraction of simulation runs for a given scenario that have achieved search success in a given number of iterations. In this context, the search success is the event in which any one of the agents comes within a prescribed detection range r_d of the hidden object. The success rate is averaged over multiple simulation runs as per equation (16). Let $n_{find}(i)$ be the number of the n_{tot} Monte Carlo iterations for which the search was successful at or before time step i. Then

$$R(i) = \frac{n_{find}(i)}{n_{tot}} \tag{16}$$

is the success rate for a given set of simulation runs. The success rate $R(i)$ demonstrates how well the swarm performs at searching the information space to find the object of interest and is thus primarily a metric for exploitation.

The distance to the search object, $J_{dist}(i)$, is a non-dimensional measure of how close the agents got to the hidden object. It is a strictly decreasing measure, in that even if the agents move farther away from the object, the measure is still based on the closest that any agent got to it up to time step i. Let $d_{min}(i)$ be the minimal distance to the object over the set of all agents up to time step i. Then the distance measure is given by

$$J_{dist}(i) = \frac{d_{min}(i)}{d_0}, \tag{17}$$

where d_0 is a reference distance for non-dimensionalization (in the examples herein, with a square $AOI\ \mathcal{W}$ with sides L, $d_0 = \sqrt{2}L$ is used as that is the longest distance possible). Note that a value of $J_{dist}(i) = 0$ implies that an agent has run into/over the object of interest and $J_{dist}(i) = 1$ implies all agents are as far as possible from the object of interest. Also note that $J_{dist}(i) \leq r_d/d_0$ for a successful search to occur, as per the definition of success used for $R(i)$ in equation (16).

The amount of the AOI covered, $J_{cov}(i)$, is a non-dimensional measure of the fraction of the AOI that has been properly covered by the agents. Since there is already a measure of $\lambda_i(x)$ as the certainty at which location x has been covered by agents, this research utilizes that term to form the coverage metric. Specifically, the coverage $J_{cov}(i)$ is given by

$$J_{cov}(i) = \int_{\mathcal{W}} \left(\frac{1-\lambda_i(x)}{A_{\mathcal{W}}}\right) dx, \tag{18}$$

where $A_{\mathcal{W}}$ is the area of the AOI (given by $A_{\mathcal{W}} = \int_{\mathcal{W}} 1\, dx$). As a practical matter, the AOI is typically represented by a tiling of cells x_j such that $\lambda_{i,j}$ is the certainty as to the coverage of cell x_j by the agents by time step i. In this case, the coverage metric has the more readily computable form

$$J_{cov}(i) = 1 - \frac{1}{N}\sum_{j=1}^{N} \lambda_{i,j}, \tag{19}$$

which is used for the computations in the sequel. It is noted that a coverage of $J_{cov}(i) = 1$ corresponds to complete coverage of the AOI, which is also saying that all cells in the region have been covered to certainty. Similarly, a coverage of $J_{cov}(i) = 0$ implies a completely entropic state where there is no certainty in the coverage.

5. NUMERICAL RESULTS

Each scenario that was examined consisted of a group of eight agents traversing a square AOI of size $L \times L$ looking for an uncertain item. Both Cooperative PSO and SPSO ran with similar settings and the results compared. Cooperative PSO was designed to take the benefits of swarm dynamics and metaheuristic optimization and focus it on autonomous control of a small set of agents. As such, the research limited the number of agents to eight. The simulation was run over 2000 iterations (time steps) and repeated for 1500 Monte Carlo iterations for each scenario.

Four environmental scenarios are considered as described in section 4 with the environmental descriptions given by the prior location expectation $\alpha(x)$ as follows:

Environment 1: $\alpha(x)$ contains a single Gaussian with $\sigma = L/5$
Environment 2: $\alpha(x)$ contains a single Gaussian with $\sigma = L/2$
Environment 3: $\alpha(x)$ contains a single Gaussian with $\sigma = L$
Environment 4: $\alpha(x)$ contains multiple Gaussians of varying sizes

For each scenario, both the SPSO and Cooperative PSO algorithms initially ran with the following two sets of parameters: $(c_I, c_P, c_G) = (1.0, 1.0, 1.5)$ and $(c_I, c_P, c_G) = (1.0, 0.5, 1.5)$.

In terms of the success rate $R(i)$, the goal is to approach $R \to 1$ as quickly as possible (with smallest i). The achieved success has been plotted at five time iterations $i = \{100, 200, 500, 1000, 2000\}$ for each set of parameters. The average is taken as in equation (16) over 1500 Monte Carlo iterations, and evaluated at each of the selected time iterations. Figure 2 shows the performance of the success rate as a comparison between Cooperative PSO and SPSO for environment 1. The two curves for each algorithm correspond to the two choices of parameter values as given previously. Note that SPSO converges much better than Cooperative PSO for this simple situation (a strong peak in the expected uncertainty of the hidden location). Also note that the achieved success rate has a very weak dependence on the specific parameters used in the algorithm.

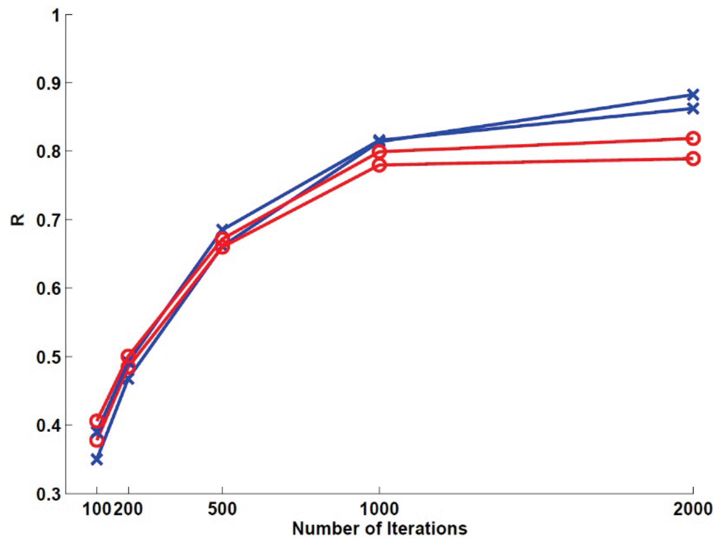


Figure 2. Success Rate R as a Function of Simulation Iterations (Time) for Environment 1

Note: In figure 2, blue curves marked with x's are for SPSO runs, while red curves marked with o's are for Cooperative PSO runs.

Figures 3 and 4 show the success rate performance for environments 2 and 3, respectively. In these plots, it is clear that Cooperative PSO has an improvement over SPSO with regard to success rate as the environment becomes more murky. That is, as the Gaussian hump becomes more diffuse (larger σ) so there is a less-precise environmental measure of where the hidden object is likely to exists. Thus, the exploration bias of Cooperative PSO allows it to achieve success at finding the object in these more-diffuse environments better than that of SPSO. Figure 5 shows the performance of the success rate for the more complex environment (environment 4).

From this plot it is seen that Cooperative PSO has a slower start than SPSO (less likely to find success early) but then performs better than SPSO as time increases. This is expected, since the SPSO behavior is to have every agent focus in towards the most likely location immediately while Cooperative PSO initially will be exploring the domain to reduce uncertainty across the space, and only then focus on heading toward the object of search.

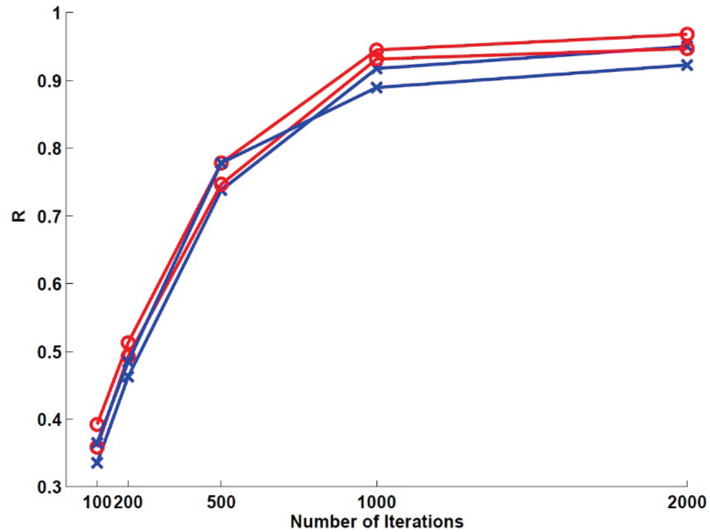


Figure 3. Success Rate R as a Function of Simulation Iterations (Time) for Environment 2

Note: In figure 3, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

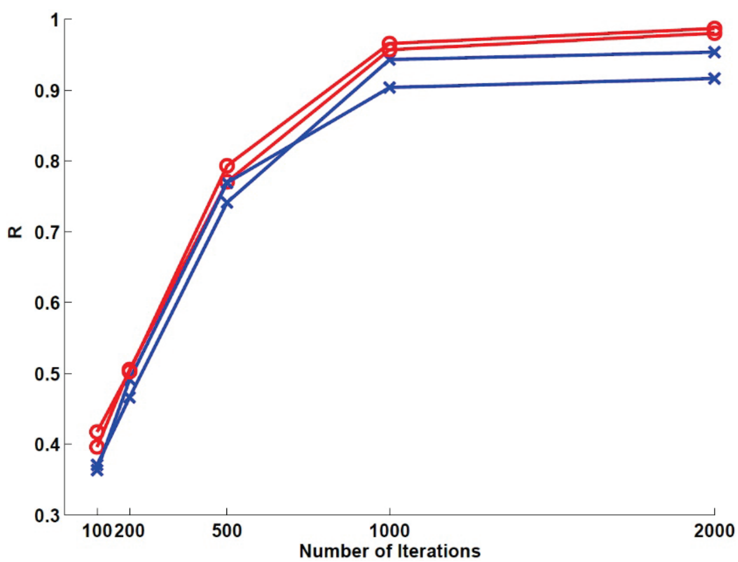


Figure 4. Success Rate R as a Function of Simulation Iterations (Time) for Environment 3

Note: In figure 4, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

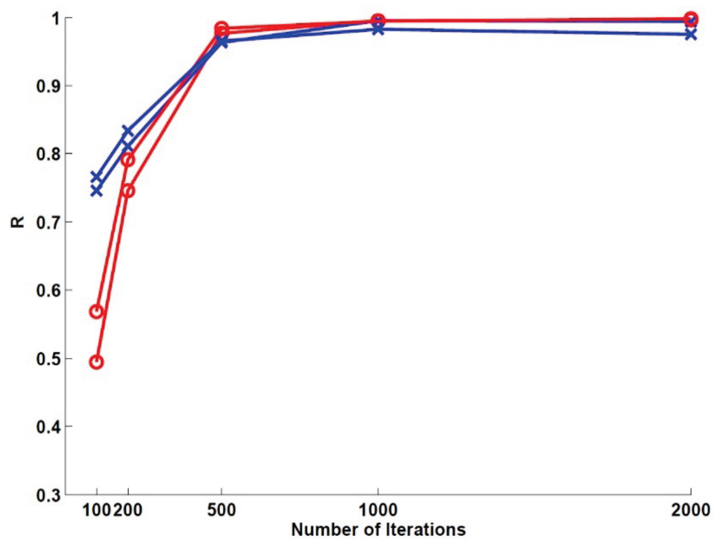


Figure 5. Success Rate R as a Function of Simulation Iterations (Time) for Environment 4

Note: In figure 5, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

The results for the success rate $R(i)$ show that the Cooperative PSO algorithm does a better job of initially exploring an uncertain search domain and then can exploit to find the object as well as PSO, if given enough iterations (time). Two additional objectives in a tradeoff plot are examined to confirm this expectation of a benefit in the seemingly conflicting performance measures of exploration versus exploitation. In particular, the amount of the AOI covered ($J_{cov}(i)$ from equation (19) is a measure of exploration, as it shows how well the group of agents have reduced uncertainty throughout the AOI. In contrast, the distance to the search object ($J_{dist}(i)$ from equation (17) is a measure of exploitation, as it shows how well the group of agents can get close to the hidden object of interest. Clearly, one would expect that exploitation comes at the expense of exploration and vice-versa, so these metrics should be in conflict.

Figure 6 shows the results of the averaged performance (averaged over the 1500 Monte Carlo iterations) of the distance metric $J_{dist}(i)$ versus the coverage metric $J_{cov}(i)$ for environment 1. The points on the plots are for iterations of $i = \{100, 200, 500, 1000, 2000\}$ with the trend moving from upper-left to lower-right as the number of iterations increases. For such a tradeoff, the most desirable location is the bottom-right corner (where the distance approaches $J_{dist}(i) \to 0$ and the coverage approaches $J_{cov}(i) \to 1$). Clearly, both algorithms lead in this direction at similar rates, although SPSO gets closer to the goal than Cooperative PSO. As the environment gets more difficult and uncertain (as shown in environments 2 and 3 in figures 7 and 8, respectively) the tradeoff begins to favor the Cooperative PSO algorithm, this being similar to what was seen in the success rate metric. Finally, as the environment becomes more complex, figure 9 (for environment 4) shows that the two algorithms lead to very similar tradeoffs as the iterations increase.

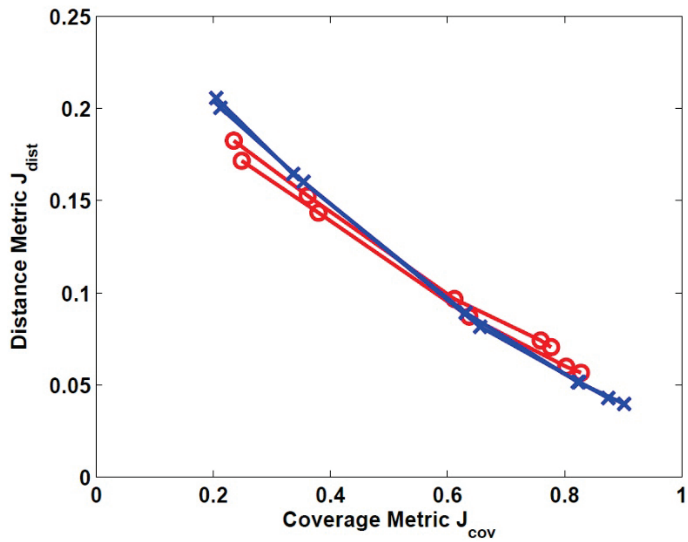


Figure 6. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 1

Note: In figure 6, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

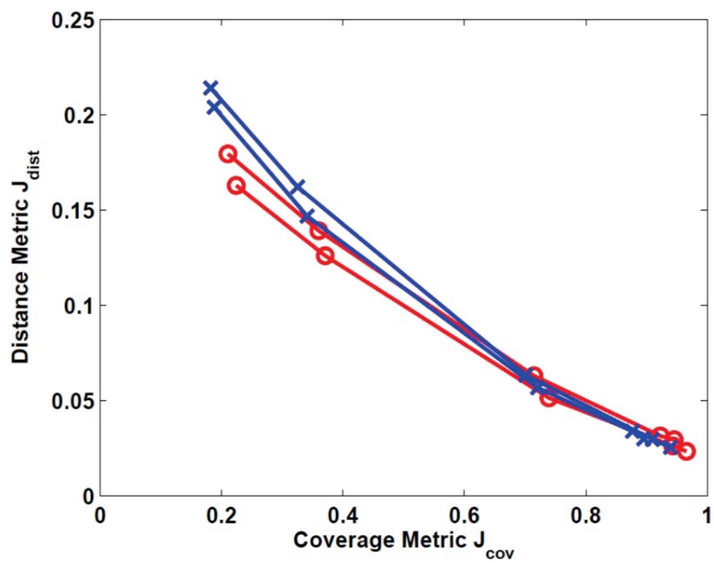


Figure 7. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 2

Note: In figure 7, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

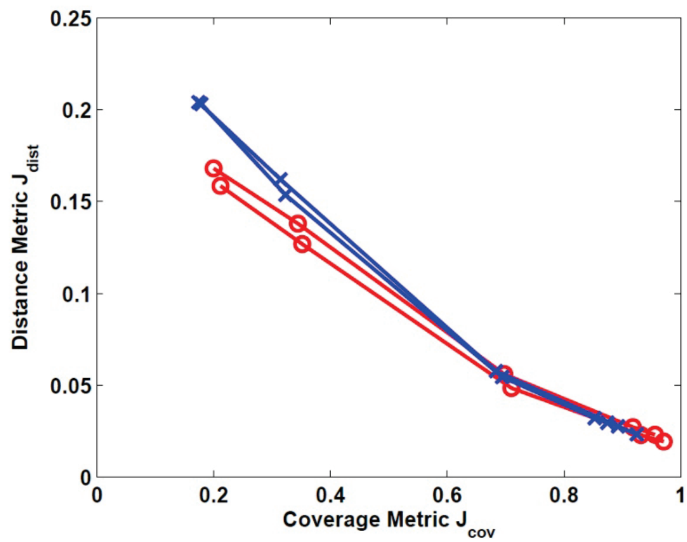


Figure 8. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 3

Note: In figure 8, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

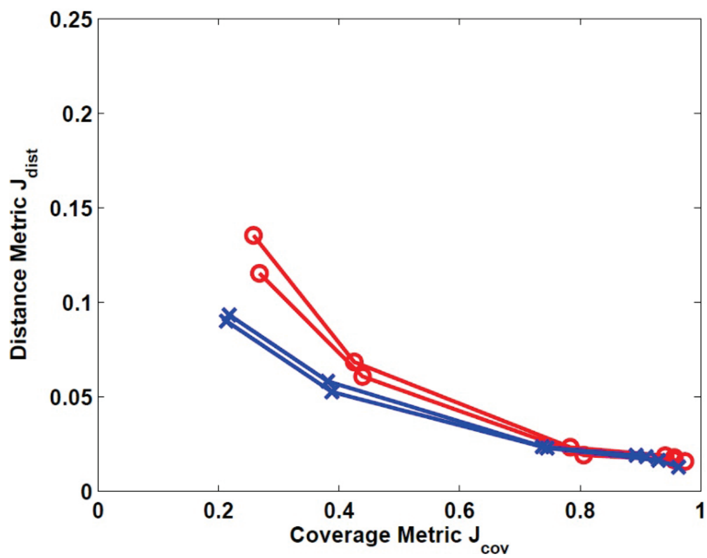


Figure 9. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 4

Note: In figure 9, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

To understand the point made by these algorithms, table 1 shows the tabulated end time results (for $i = 2000$ iterations) for each case. Table 1 further shows that the parameter choice of $c_P = 0.5$ is preferable to $c_P = 1.0$ for the Cooperative PSO algorithm for all scenarios (preferable since it leads to higher J_{cov} and lower J_{dist}). However, for the SPSO algorithm the parameter choice of $c_P = 1.0$ is preferable to $c_P = 0.5$ for all scenarios. Thus, it is concluded that the Cooperative PSO algorithm has a larger dependence on the group bias effect (relative to the personal bias effect) when compared to the SPSO algorithm. This bias towards the group allows Cooperative PSO to do a better job of exploration (i.e., coverage) without foregoing much performance in exploitation (i.e. distance). This is a clear advantage of the Cooperative PSO algorithm.

Since the Cooperative PSO algorithm benefits from the group advantage, and the group advantage is more pronounced when there is low uncertainty in the region (corresponding to low λ), this research seeks to exaggerate that effect by dynamically weighting the personal bias term. In particular, when there is large uncertainty in the region (corresponding to high λ), it is not expected that the algorithm benefit from the group components, and thus it is desirable to bias more towards the individual. Similarly, when there is low uncertainty in the region, it is desirable to bias more towards the group. To accomplish that, this research proposes a modification to Cooperative PSO that uses the local value of $\lambda(x)$ (in particular, using the $\lambda(x)$ that corresponds to the agent's current x-location) as a dynamic weighting c_P for the personal bias term. To fairly compare the benefit of this dynamic value of $c_P = \lambda$, the same dynamic parameter is applied to both the Cooperative PSO and SPSO, and a new set of simulations is run in each of the four environments. The resulting tradeoff curves of J_{dist} versus J_{cov} for each of the environments are shown in figures 10 through 13. From these tradeoff curves, it is clear that Cooperative PSO can take advantage of this dynamic weighting parameter and achieve a better tradeoff than SPSO in all four environment scenarios. To specifically show the benefit over the static weights, the final (iteration $i = 2000$) values are included for both metrics in each environment in table 2.

This dynamic weighting replacing the static weighting of Cooperative PSO's personal history component makes for a swarm algorithm of the same structure as PSO with a much better multi-agent search performance in both the exploration and exploitation metrics.

Table 1. Comparison of Performance of SPSO and Cooperative PSO on Example Problems with Constant Values of c_P

		J_{dist}		J_{cov}	
		SPSO	Cooperative PSO	SPSO	Cooperative PSO
$c_P = 1.0$	Environment 1	0.040	0.070	0.90	0.78
	Environment 2	0.025	0.030	0.94	0.95
	Environment 3	0.023	0.023	0.92	0.96
	Environment 4	0.013	0.018	0.96	0.96
$c_P = 0.5$	Environment 1	0.043	0.057	0.87	0.83
	Environment 2	0.030	0.023	0.90	0.97
	Environment 3	0.030	0.019	0.87	0.97
	Environment 4	0.018	0.016	0.91	0.97

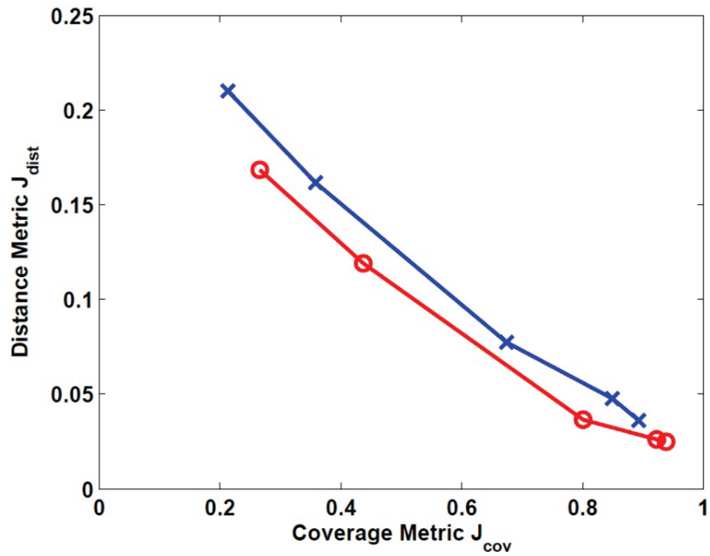


Figure 10. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 1 with a Dynamic c_P

Note: In figure 10, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

Table 2. Comparison of Performance of SPSO and Cooperative PSO on Example Problems with a Dynamic Value of c_P

		J_{dist}		J_{cov}	
		SPSO	Cooperative PSO	SPSO	Cooperative PSO
$c_P = \lambda$	Environment 1	0.036	0.025	0.89	0.94
	Environment 2	0.032	0.017	0.88	0.98
	Environment 3	0.033	0.018	0.85	0.98
	Environment 4	0.019	0.015	0.89	0.99

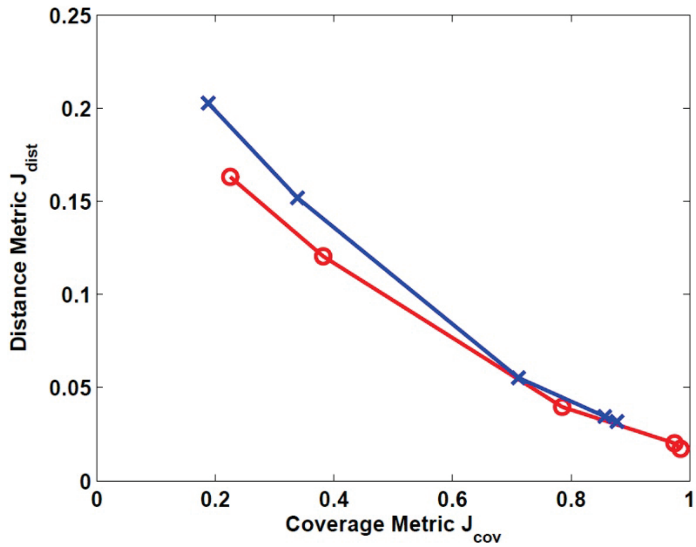


Figure 11. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 2 with a Dynamic c_P

Note: In figure 11, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

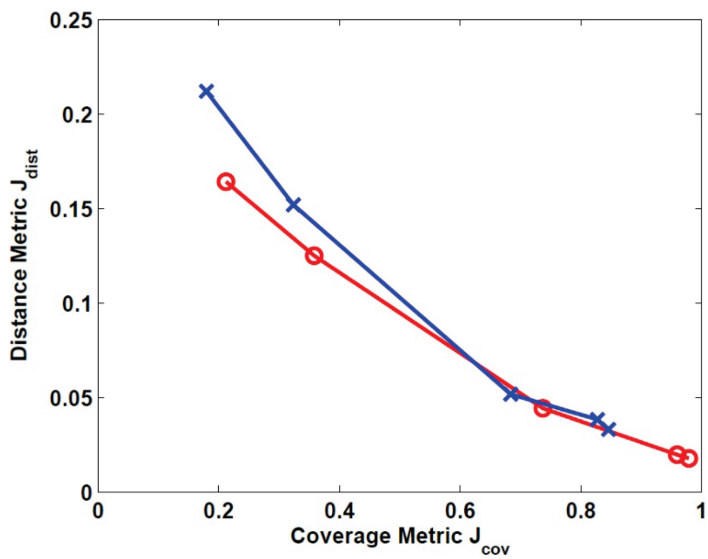


Figure 12. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 3 with a Dynamic c_P

Note: In figure 12, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

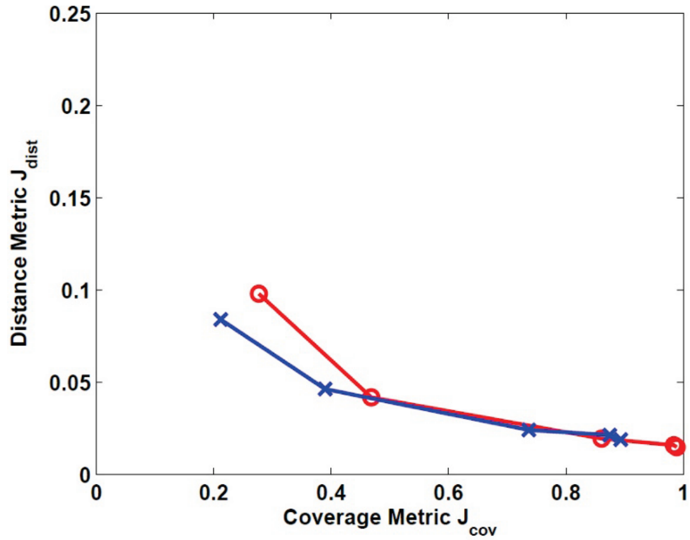


Figure 13. Tradeoff of Distance Metric J_{dist} versus Coverage Metric J_{cov} for Environment 4 with a Dynamic c_P

Note: In figure 13, blue curves marked with x's are for SPSO runs while red curves marked with o's are for Cooperative PSO runs.

6. CONCLUSION

This report describes the development of a new autonomous swarm control algorithm, called Cooperative Particle Swarm Optimization (Cooperative PSO) that is inspired by a well-known swarm optimization technique, PSO. The use of a dynamic measure of uncertainty in the environment's information space has been developed as a fundamental mechanism in the Cooperative PSO algorithm. This report also provides the results of a set of simulation experiments that demonstrate the effectiveness of Cooperative PSO compared to a standard PSO (SPSO) implementation when using a small number of agents to perform a cooperative search. This distributed multi-agent control procedure performs better than the PSO-based approach when measured in both an exploration and exploitation metric, and is especially beneficial in managing the tradeoff between these performance goals. Future research into this approach involves the test and evaluation of Cooperative PSO on real-world robotics systems.

REFERENCES

1. M. G. Hinchey, R. Sterritt, and C. Rouff, "Swarms and swarm intelligence," *Computer*, vol. 40, pp. 111–113, April 2007.
2. L. D. Silva and H. Ekanayake, "Behavior-based robotics and the reactive paradigm: A survey," *Proceedings of International Workshop in Data Mining and Artificial Intelligence (DMAI '08)*, pp. 36–43, December 2008.
3. A. Turing, "Computing machinery and intelligence," *Mind, New Series*, vol. 59, pp. 433–460, October 1950.
4. A. Tews, "Adaptive multi-robot coordination for highly dynamic environments," *International Conference on Computational Intelligence for Modelling (CIMCA)*, 2001.
5. M. Koes, I. Nourbakhsh, and K. Sycara, "Heterogeneous multirobot coordination with spatial and temporal constraints," *AAAI National Conference on Artificial Intelligence*, 2005.
6. K. DeMedeiros, R. Balasubramanian, C. Duarte, and G. Martel, "A novel token passing algorithm for management of refueling vehicles in a multi-UUV 24/7 mission scenario," *Oceans 2010 MTS/IEEE Seattle*, pp. 1–7, September 2010.
7. K. DeMedeiros and R. Balasubramanian, "Task management using token passing for a group of cooperating unmanned undersea vehicles," *UUST 2011*, June 2011.
8. C. Tye, M. Kinney, J. Frenzel, M. O'Rourke, and D. Edwards, "A MOOS module for autonomous underwater vehicle fleet control," *Oceans 2009*, pp. 1–6, October 2009.
9. D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, pp. 65–85, 1994.
10. V. Maniezzo, A. Colorni, and M. Dorigo, "Distributed optimization by ant colonies," *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pp. 134–142, MIT Press, 1992.
11. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
12. J. Kennedy and R. Eberhart, "Particle swarm optimization," *Proceedings IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.
13. D. Karaboga, "An idea based on honey bee swarm for numerical optimization," *Technical Report Tr06, Erciyes University*, October 2005.
14. K. N. Krishnanand and D. Ghose, "Detection of multiple source locations using a glowworm metaphor with applications to collective robotics," *Proceedings 2005 IEEE Swarm Intelligence Symposium, SIS 2005*, pp. 84–91, June 2005.
15. E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232 – 2248, Special Section on High Order Fuzzy Sets, 2009.

16. K. DeMedeiros and T. Wettergren, “Cooperative swarm optimization: A teamwork-based approach to reduction of uncertainty,” *OCEANS 2016 MTS/IEEE Monterey*, pp. 1–6, September 2016.

17. R. Hassan, B. Cohanim, O. de Weck, and G. Venter, “A comparison of particle swarm optimization and the genetic algorithm,” *Proceedings 46th AIAA/ASME/AHS/ASC Structures, Structual Dynamics & Materials Conference*, 2005.

18. M. Iruthayarajan, S. Willjuice, and S. Baskar, “Optimization of PID parameters using genetic algorithm and particle swarm optimization,” *Information and Communication Technology in Electrical Sciences (ICTES 2007), ICTES. IET-UK International Conference on*, pp. 81–86, December 2007.

19. Q. Zhu, Q. Lin, W. Chen, K. C. Wong, C. A. C. Coello, J. Li, J. Chen, and J. Zhang, “An external archive-guided multiobjective particle swarm optimization algorithm,” *IEEE Transactions on Cybernetics*, vol. 47, pp. 2794–2808, September 2017.

20. W. Dong and M. Zhou, “A supervised learning and control method to improve particle swarm optimization algorithms,” *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. PP, no. 99, pp. 1–14, 2016.

21. K. Chiba, “Performance comparison of evolutionary algorithms applied to hybrid rocket problem,” *The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems*, pp. 1673–1678, November 2012.

22. W. Shyr, “Introduction and comparison of three evolutionary-based intelligent algorithms for optimal design,” *Convergence and Hybrid Information Technology, 2008. ICCIT ‘08. Third International Conference on*, vol. 2, pp. 879–884, November 2008.

23. B. Niu, H. Huang, L. Tan, and Q. Duan, “Symbiosis-based alternative learning multi-swarm particle swarm optimization,” *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 14, pp. 4–14, January 2017.

24. T. P. Hong, J. C. Chen, W. Y. Lin, and C. H. Chen, “Analysis of parallel sub-swarm PSO with the same total particle numbers,” *2015 18th International Conference on Network-Based Information Systems*, pp. 616–619, September 2015.

25. Y. Wang and Z. Cai, “A hybrid multi-swarm particle swarm optimization to solve constrained optimization problems,” *Frontiers of Computer Science in China*, vol. 3, no. 1, pp. 38–52, 2009.

26. A. Atyabi, S. Phon-Amnuaisuk, and C. K. Ho, “Applying area extension PSO in robotic swarm,” *Journal of Intelligent and Robotic Systems*, vol. 58, pp. 253–285, June 2010.

27. M. Bonyadi and Z. Michalewicz, “Particle swarm optimization for single objective continuous space problems: A Review,” *Evolutionary Computation (corrected proof)*, 2017.

DISTRIBUTION LIST

Internal

Codes:	1033	(Corporate Research and Information Center (CRIC))
	8512	(K. DeMedeiros (2))
	00T1	(T. Wettergren (2))

External

Defense Technical Information Center (DTIC)

Total: 6