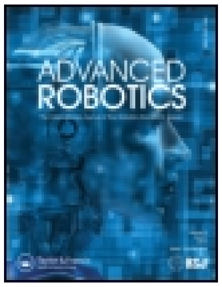

Advanced Robotics
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tadr20

A PSO multi-robot exploration approach over unreliable MANETs

Micael S. Couceiro [a] [b] , Rui P. Rocha [a] & Nuno M.F. Ferreira [b]
[a] Institute of Systems and Robotics, University of Coimbra , Pólo II, Coimbra , 3030-290 , Portugal
[b] RoboCorp, Engineering Institute of Coimbra , Quinta da Nora, Coimbra , 3030-199 , Portugal

Published online: 18 Jul 2013.

To cite this article: Advanced Robotics (2013): A PSO multi-robot exploration approach over unreliable MANETs, Advanced Robotics

To link to this article: http://dx.doi.org/10.1080/01691864.2013.819605

Advanced Robotics, 2013
http://dx.doi.org/10.1080/01691864.2013.819605

FULL PAPER

A PSO multi-robot exploration approach over unreliable MANETs

Micael S. Couceiro[a,b]*, Rui P. Rocha[a] and Nuno M.F. Ferreira[b]

[a]*Institute of Systems and Robotics, University of Coimbra, Pólo II, Coimbra 3030-290, Portugal;* [b]*RoboCorp, Engineering Institute of Coimbra, Quinta da Nora, Coimbra 3030-199, Portugal*

(*Received 24 October 2012; accepted 25 February 2013*)

This paper proposes two extensions of Particle Swarm Optimization (PSO) and Darwinian Particle Swarm Optimization (DPSO), respectively denoted as RPSO (Robotic PSO) and RDPSO (Robotic DPSO), so as to adapt these promising biologically inspired techniques to the multi-robot systems domain, by considering obstacle avoidance and communication constraints. The concepts of social exclusion and social inclusion are used in the RDPSO algorithm as a 'punish–reward' mechanism, thus enhancing the ability to escape from local optima. Experimental results obtained in a simulated environment shows the superiority of the RDPSO evidencing that sociobiological inspiration can be useful to meet the challenges of robotic applications that can be described as optimization problems (e.g. search and rescue). Moreover, the performance of the RDPSO is further evaluated within a population of up to 12 physical robots under communication constraints. Experimental results with real platforms show that only 4 robots are needed to accomplish the herein proposed mission and, independently on the number of robots and maximum communication distance, the global optimum is achieved in approximately 90% of the experiments.

Keywords: swarm robotics; distributed search; MANET; initial deployment

1. Introduction

Behavior-based paradigms have a strong influence in multi-robot system research. Moreover, the analysis of the social characteristics of insects and animals is essential in order to apply these findings to the design of multi-robot systems (MRS). The most common application of this knowledge is in the use of simple local control rules of several biological societies (e.g. ants, bees and birds) to the development of similar behaviors in cooperative MRS.[1] The Particle Swarm Optimization (PSO) developed by Kennedy & Eberhart [2] is an optimization technique that models a set of potential problem solutions as a swarm of particles moving around in a virtual search space. However, a general problem with the PSO and other optimization algorithms is that of becoming trapped in a local optimum, such that it may work in some problems but may fail on others. In search of a better model of natural selection using the PSO algorithm, the Darwinian Particle Swarm Optimization (DPSO) was formulated by Tillett et al. [3]. In this algorithm, multiple swarms of test solutions, performing just like an ordinary PSO, may exist at any time with some rules governing the collection of swarms that are designed to simulate natural selection. Just like in MRS where groups of robots interact to accomplish their goals,[4] both PSO and DPSO use groups of interacting virtual agents (i.e. particles) in order to achieve their optimization. However, contrarily to virtual agents, robots are designed to act in the real world where obstacles need to be taken into account. Also, and since that in certain environments or applications, such as hostile environments, search and rescue, disaster recovery, battlefields, space and others, the communication infrastructure may be damaged or missing, and the self-spreading of autonomous mobile nodes of a mobile *ad hoc* network (MANET) over a geographical area needs to be considered. For instance, the work of Sahin et al. [5] presented a few bio-inspired computation techniques to carry out real missions such as military applications. Under those circumstances, the authors focused on the self-spreading of autonomous mobile nodes over an unknown geographical area so as to create and maintain the MANET connectivity.

Bearing these ideas in mind, this article proposes extensions of both PSO and DPSO to MRS, respectively denoted as RPSO (Robotic PSO) and RDPSO (Robotic DPSO), which takes into account obstacle avoidance and communication constraints, thus guaranteeing the

*Corresponding author. Email: micaelcouceiro@isr.uc.pt
This paper was originally submitted to the special issue on 'Safety, Security and Rescue Robotics' of *Advanced Robotics*, Vol. 27 Issue 5 (2013).

MANET connectivity. In order to establish the initial deployment of robots while preserving the MANET connectivity, this paper also proposes a novel approach inspired on the *Spiral of Theodorus*. The algorithm is demonstrated in multi-robot exploration tasks, wherein each robot is represented by a particle that needs to be evaluated at each iteration. After each set of evaluations, robots communicate to share the objective information (e.g. cost or fitness) needed to progress to the next iteration of the algorithm while avoiding obstacles and fulfilling MANET connectivity.

2. Related work

In the last two decades, a significant progress in applied computing and robotics has occurred through the application of principles derived from the study of biology. The navigation of groups of robots, especially swarm robots, has been one of the fields that has benefited from biological inspiration.[6] However, real MRS present several constraints that need to be considered. For instance, communication constitutes one of the most important resources for more effective cooperation among robots and improved robust collective performance. The development of robot teams for surveillance or rescue missions requires that robots have to be able to maintain communication among them without the aid of a communication infrastructure. Besides that, robots also need to ensure MANET connectivity in order to explicitly exchange information within multi-hop network paths, thus not restricting unnecessarily the team's range.

One of the first adapted versions of the PSO to handle real-world constraints such as obstacles is presented by Min et al. [7]. Similar to the current work, this approach adjusts the velocity and direction of the mobile robot in real time thus allowing the robot to reach its goal, avoiding obstacles in the way. Each robot runs an entire swarm and the global best particle is considered the best solution. Unfortunately, simulation results presented the comparison with Artificial Potential Field algorithms with only one robot. Also, simulation experiments lack some information such as the distance the robot needs to travel (since the time which the mobile robot spends in reaching the goal is presented).

Another similar approach was developed by Pugh and Martinoli [8] where an adapted version of the PSO to distributed unsupervised robotic learning in groups of robots with only local information is presented. The main difference between this algorithm and classical PSO is that each robot (i.e. particle) only takes into consideration the information of the robots within a fixed radius r (omnidirectional communication). The authors analyzed how the performance was affected if the standard PSO neighborhood structure was adapted to a more closely model, which is possible in a real robot group with limited communication abilities. Experimental results obtained using *Webots* simulator showed that the adapted version of the PSO maintained good performance for groups of robots of various sizes when compared to other bio-inspired methods. However, contrarily to the presented RDPSO algorithm proposed herein, all bio-inspired methods used, including the adapted PSO, tend to get trapped in local solutions. Furthermore, and contrarily to the experimental results shown in this work, the authors do not use multi-hop connectivity and do not apply any kind of algorithm to enforce communication between robots. Similarly, Hereford and Siebold [9] presented an embedded version of the PSO in swarm platforms. As in RDPSO, there is no central agent to coordinate the robots movements or actions. Despite the potentialities of the physically embedded PSO, experimental results were carried out using a population of only three robots, performing a distributed search in a scenario without local solutions. Also, collision avoidance and fulfillment of MANET connectivity were not considered. To maintain the MRS connectivity, a behavior-based strategy was presented by Arrichiello et al. in [10]. The authors presented the extension of the Null-Space-based Behavioral approach to control a group of marine vehicles to execute multi-robot missions, such as formation control and cooperative target visiting with communication constraints. This is a promising approach since it would be possible to merge the behaviors of the proposed RDPSO with different priorities, in order to define the final motion directives of the robots. However, the design choices concerning how to organize the behaviors in priority represents a higher complexity, since these choices derive from practical considerations related to both the mission objective and the hardware/software features of the robotic system.

Çelikkanat and Sahin studied the flocking behavior to steer self-organized flocks in both physical and simulated mobile robots.[11] Experimental results showed that the flock could be steered along a desired direction maintaining a ratio of informed robots above 10% of the population. Despite the dissimilarities of the main objective when compared to the scope of this article, the proposed RDPSO algorithm also tries to reduce the shared information between the robots, since it divides the population in multiple swarms. This means that if a population is divided into three swarms, the exchanged information between robots of the same swarm will be one-third of the whole information. Tardioli et al. proposed a robots' navigation based on a Spring-Damper Systems (SDSs), with one robot being the leader of the formation and other robots being the slaves.[12] This kind of approach incorporates the management of the system dynamics in real situations dealing with dynamic behavior of robots. Also, the SDS mechanism allows maintaining multi-hop routes between nodes of sufficient quality, in order to avoid the network becoming disconnected. Thus, a measure of the communication link quality is used [13] instead of the commonly used communication range, such as [14]. Like in our approach the robots movements are restricted if necessary by using this measure. However, the use of SDSs introduces constraints that traditional allocation methods do not face. Similar and simpler methodologies that also take into consideration the dynamic behavior of robots could be used such as elastic bands [15] or even fuzzy systems.[16]

3. Robotic PSO

In nature, some complex group behaviors arise in biological systems composed of swarms that are observed in a variety of simple social organisms (e.g. ants, bees).[1] One of the most relevant topics in MRS is the modeling and control of the population. The PSO [2] consists of a number of N particles that collectively move on the search space in search of the global optimum. Each particle is characterized by its position and performance $f(x_n[t])$ at each discrete time, or iteration, $t \in \mathbb{N}$. In each step of the algorithm, an objective function is used to evaluate the particle success. The cost of a particle closer to the global solution is lower than that of a particle that is farther. Conversely, the fitness of a particle closer to the global solution is higher than that of a particle that is farther. PSO thrives to minimize a cost function, or maximize a fitness function. To model the swarm, each particle n moves in a multidimensional space according to position vector ($x_n[t+1]$) and velocity vector ($v_n[t+1]$), which are highly dependent on local best vector ($\breve{x}_n[t]$) and global best vector ($\breve{x}_n[t]$) and global best vector ($\breve{g}_n[t]$) information. The size of the vectors depends on the dimension of the multidimensional space.

$$v_n[t+1] = wv_n[t] + \rho_1 r_1(\breve{x}_n[t] - x_n[t]) + \rho_2 r_2(\breve{g}_n[t] - x_n[t]) \quad (1)$$

$$x_n[t+1] = x_n[t] + v_n[t+1] \quad (2)$$

Coefficients w, ρ_1 and ρ_2 assign weights to the inertial influence, the local best and the global best when determining the new velocity, respectively. Typically, the inertial influence is set to a value slightly less than 1. ρ_1 and ρ_2 are constant integer values, which represent 'cognitive' and 'social' components. Depending on the application and the characteristics of the problem being considered, tuning these parameters properly will lead to better results. Parameters r_1 and r_2 are random vectors, wherein each component is generally a uniform random number between 0 and 1. The intent is to multiply a new random component per velocity dimension, rather than multiplying the same component with each particle's velocity dimension. In the beginning, i.e. $t=0$, the particles' velocities are set to zero and their position is randomly set within the boundaries of the search space. The local and global bests are initialized with the worst possible values, taking into account the nature of the problem. The stopping criteria (e.g. number of iterations) also needs to be adjusted to get overall good solutions in acceptable time. The RPSO, just like the PSO, basically consists on a population of robots that collectively move on the search space (e.g. catastrophic scenario, city) in search of the global optimum (e.g. number of victims, number of passengers); each robot is characterized by its pose and performance. For instance, if we have a group of mobile olfactory robots that are trying to find a gas leak (cf. [17]) in an indoor environment, each robot will be characterized by its pose (i.e. position and orientation) and by the corresponding value of gas density. The main difference between RPSO and PSO resides in the implementation of real-world scenarios where strategies for obstacle avoidance and communication constraints need to be taken into account.

3.1. Obstacle avoidance

RPSO algorithm tries to minimize a cost function, or maximize a fitness function depending on the mission objective. The approach proposed in this section seeks to create a new cost or fitness function in such a way that it would guide the robot to perform the main mission while avoiding obstacles. When a robot must move from any arbitrary start position to any target position in the environment, it must be able to avoid both static and dynamic obstacles.[18] For this purpose, we assume that each robot is equipped with sensors capable of sensing the environment for obstacle detection within a finite sensing radius r_s. A monotonic and positive sensing function $g(x_n[t])$ is defined. This function depends on the sensing information, i.e. distance from the robot to obstacle. Note that, in most situations, the sensing function $g(x_n[t])$ can be represented as the relation between the analog output voltage of distance sensors and the distance to the detected object.

Let us first rewrite Equations (1) and (2) as the discrete matrix equations considering iteration t:

$$V[t+1] = I[t] + \rho_1 r_1 C[t] + \rho_2 r_2 S[t] \quad (3)$$

$$X[t+1] = X[t] + V[t+1] \quad (4)$$

wherein each line of matrices $V[t]$ and $X[t]$ represents the velocity and position vector of each different robot, respectively. Matrices $I[t]$, $C[t]$ and $S[t]$ represent the inertial, cognitive and social matrix components, respectively. Then, the new velocity of each robot can be defined as an extension of Equation (3) taking into account the presence of obstacles:

$$V[t+1] = I[t] + \rho_1 r_1 C[t] + \rho_2 r_2 S[t] + \rho_3 r_3 O[t] \tag{5}$$

where ρ_3 and r_3 are the obstacle susceptibility weight and respective random vector, while $O[t]$ is the obstacle matrix represented by the difference between the position of each robot that optimizes the monotonically decreasing or increasing *sensing function* $g(x_n[t])$ and its current position. In other words, when a robot does not sense any obstacle at time t, the best position that optimizes $g(x_n[t])$ is constantly updated and equal to the current position $x_n[t]$. Afterward, if the robot detects any obstacle inside its sensing range, the best position that optimizes $g(x_n[t+1])$ is not updated, thus creating a repulsive force toward its last best position $x_n[t]$.

3.2. Communication constraints

It has generally been assumed in MRS that each robot has the ability to communicate with any other robot with small consideration for the quality and performance of the wireless communication network. Although being valid in particular situations, such an assumption does not generally hold. Since robots may move apart to further areas, it is important to have a pervasive networking environment for communications among robots. Furthermore, without a preexistent infrastructure, robots need to be able to act as intermediate nodes, i.e. routers, in order to relay information from one point to another, thus supporting multi-hop communication in a MANET.[19,20]

3.2.1. Problem statement

Consider a population of N robots where each robot is both an exploring agent of the environment and a mobile node of a MANET that performs packet forwarding, according to a paradigm of *multi-hop communication*. The goal is to ensure that the robots explore an unknown environment, while ensuring that the MANET remains connected throughout the mission.

3.2.2. General approach

The connectivity between robots can be described by means of a *link matrix* $L=\{l_{ij}\}$ for an N-node network, where each entry represents the link between nodes (i.e. robot) i and j. The link is defined accordingly with the users' preferences. The most common approaches include: (1) calculating the l_{ij} values as functions of the distance between pairs of nodes and indicating the *link distance* between them [14]; (2) calculating the l_{ij} values as functions of the radio quality signal between pairs of nodes indicating the *link quality* between them.[13] The approach 1 is the most common and easier to implement in simulation, since the link between nodes is always symmetric, i.e. the distance between robot i and j is the same as the distance between robot j and i. However, in approach 2, the link between nodes can be asymmetric, i.e. the radio signal received by robot i when robot j transmits can be different from the one received by robot j when robot i transmits. In general, the differences are small and it is assumed that the link matrix is always square and symmetric. Nevertheless, in real experiments, the signal quality is easier to obtain than the distance between robots since most part of wireless equipment benefit from the Received Signal Strength Indicator.[21]

Depending on the chosen approach (1 or 2), an *adjacency matrix* $A=\{a_{ij}\}$ can be defined based on the maximum distance or minimum radio quality signal between nodes, respectively.[19] The adjacency matrix, i.e. one-hop connectivity matrix, where a 1 entry at (i, j) indicates a connection between node i and j and a 0 entry at (i, j) indicates no connection between node i and j, represents the neighbors of each node, i.e. direct connection between robots.

$$a_{ij} = \begin{cases} 1, & i \text{ and } j \text{ connected} \\ 0, & i \text{ and } j \text{ not connected} \end{cases} \tag{6}$$

Note that the diagonal elements (i.e. when $i=j$) of the adjacency matrix are set equal to 0. If the communication system supports the relay of messages to distant nodes via intermediate nodes, then multi-hop connections can be made. In the case where each robot corresponds to a node, in order to overcome the non-connectivity between them, the desired position of each robot, i.e. $x_n[t+1]$, must be controlled, since it influences the link matrix. One way to ensure the full connectivity of the MANET is to 'force' each robot to communicate with its nearest neighbor that has not chosen it as its nearest neighbor. Since the connectivity depends on the distance/signal quality, connectivity between nodes may be enforced by computing the minimum/maximum values of each line of adjacency matrix A, after excluding zeros and (i, j) pairs previously chosen. Therefore Equation (5) can be rewritten as:

$$V[t+1] = I[t] + \rho_1 r_1 C[t] + \rho_2 r_2 S[t] + \rho_3 r_3 O[t] + \rho_4 r_4 M[t] \tag{7}$$

where ρ_4 and r_4 are the communication enforcing weight and respective random vector, while $M[t]$ is the MANET matrix that is represented by the difference between the positions of the nearest neighbor increased by the maximum allowed communication range $d_{\max}$ toward robot's current position . In this work, the multihop connectivity matrix $C^{(N-1)}$ and auxiliary matrices (C_B and C_{break}) will only be used as information about network topology.

To further understand matrix $M[t]$ consider the topology depicted in Figure 1. As it may be perceived, robot 2 is the nearest neighbor of robot 1 and is at the

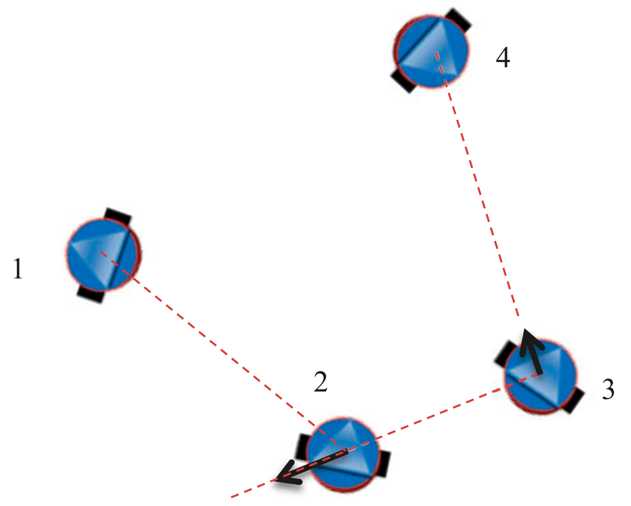

Figure 1. Illustration of a MANET topology of a swarm. Dashed lines represent the maximum distance $d_{\max}$ between each pair of robots and the arrows represent the force vectors that ensure MANET connectivity.

correct distance $d_{\max}$ resulting in a null force connectivity vector. The nearest neighbor of robot 2 is robot 3 which is too close, thus resulting in a repulsive force at robot 2 in order to ensure $d_{\max}$. Finally, the nearest neighbor of robot 3, that was not previously chosen, is robot 4 which is too far away, thus being affected by an attraction force toward robot 4.

The exchanged data concerning to the signal quality or robot's position will allow the implicit processing of the enforcing network connectivity algorithm by the team in a distributed way. In other words, every robot needs to be aware of the position or signal quality of all other robots in the swarm in order to compute the enforcing network connectivity algorithm. This is an algorithm limitation since all robots need to be equipped with localization systems (e.g. GPS). An alternative to it would be extending the GPS capabilities of some robots to non-GPS robots [22] using strategies to find the teammates position under their visual range. For instance, if robots are equipped with laser range finders the use of retro-reflective markers can be used for recognition. Since the implementation of such strategies is out of scope of this paper they will not be taken into account.

One of the major concerns in this approach is that all robots should have an initial deployment that preserves the communication between the robots in the population. Moreover, it is also known that in classical PSO algorithms particles need to be scattered throughout the scenario. Hence, next section presents a novel methodology to establish an initial planar deployment of the robots that preserves the connectivity of the MANET, while spreading out the robots as much as possible.

3.3. Initial deployment

One of the fundamental problems in MRS that has not been fully addressed is how to deploy a group of robots over an environment to carry out sensing, surveillance, data collection, or distributed servicing tasks.[23] For instance, when the robots are transported to the catastrophe site, they need to be deployed. The deployment problem is to decide how many robots and where they will be initially located before performing the SR mission using their control strategy (e.g. coverage) (e.g. [24]). One of the common approaches in the initial deployment of mobile robots is using a random distribution along the scenario.[25] This methodology is the simplest way for deploying robots and in most situations (e.g. SR), the distribution of the points of interest (e.g. victims) is random. However, in real situations, it is necessary to ensure several constraints of the system. If the network supports multi-hop connectivity, this kind of constraints may significantly increase the complexity of the random distribution since it would depend not only on the communication constraints but also on the number of robots and their own position. Moreover, random deployment may cause unbalanced deployment therefore increasing the hardware cost (e.g. number of needed robots, energy depletion).

This work tries to take advantage of a random planar deployment of robots, while eliminating the disadvantages inherent to it and considering the communication constraints by using a deployment strategy based on the *Spiral of Theodorus*. This spiral is composed of contiguous right triangles, formerly called rectangled triangles, with each cathetus having a length equal to 1.[26] Triangles' hypotenuses h_i are given by the square root to a consecutive natural number, with $h_1 = \sqrt{2}$. The use of the spiral of Theodorus to carry out the initial deployment of robots, requires two adjustments: (1) the initial position of each robot is set at the further vertex of the center of the spiral for each right triangle with a random orientation; and (2) the size of the cathetus is set as the maximum communication range $d_{\max}$ consequently changing the triangles' hypotenuses h_i to the product between the maximum communication range $d_{\max}$ and the square root of the consecutive natural number. These assumptions allow having an initial deployment of the robots in the target area which depends on both the number of robots and the communication constraints (Figure 2(a)).

The growth of the angle φ_k of the next triangle (or spiral segment) k, can be calculated using the trigonometric properties of right triangles.

$$\varphi_k = \arctan \frac{1}{\sqrt{k}} \tag{8}$$

The total angle φ_n for the nth robot is calculated as the cumulative sum presented above.

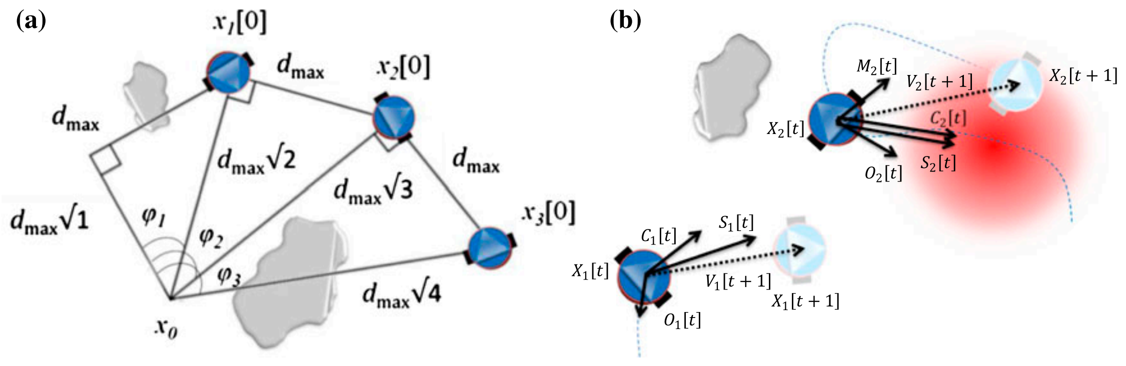

Figure 2. (a) Initial deployment based on the *spiral of Theodorus* of a swarm of robots. (b) Geometrical illustration of the RPSO.

$$\varphi_n = \sum_{k=1}^{n} \varphi_k \tag{9}$$

Once again, using the trigonometric functions, the initial planar position (i.e. $t=0$) of each robot n can easily be calculated.

$$x_n[0] = x_o + \begin{bmatrix} d_{\max}\sqrt{n+1} \times \cos(\varphi_n) \\ d_{\max}\sqrt{n+1} \times \sin(\varphi_n) \end{bmatrix} \tag{10}$$

where x_0 is the center of the spiral which can be randomly assigned at each trial ensuring the efficiency of the stochastic algorithms. Hence, the initial deployment of a swarm will correspond to a spiral in which the position of each robot depends on the prior deployed robot and the center of the spiral x_0. To allow the autonomous deployment of robots in a scenario, a preprocessing of the environment needs to be undertaken in order to prevent robots from being deployed into areas of no interest (e.g. water, obstacles, and other robots). This can be accomplished with unmanned aerial vehicles through image segmentation (cf. [22]).

In short, all the previously described matrices can be represented within a small population in Figure 2(b). Merging all these features, the RPSO algorithm can be summarized as shown in Algorithm 1.

This section bridged the gap between classical PSO algorithms and its application in MRS, denoted as RPSO. However, both PSO and RPSO have an important drawback: they may get stuck on local optima. In other words, the RPSO do not have any mechanism to avoid stagnation nor to adapt the behavior of robots based on contextual information.

Next section proposes a novel multi-robot exploration algorithm inspired by the DPSO formulated by Tillet et al. [3] which overcomes the problem.

Algorithm 1: RPSO Algorithm

```
1   Initialize swarm
2   Initialize $\boldsymbol{X}[t]$ using the spiral of Theodorus
    methodology (Section 3.3)
3   Initialize $\boldsymbol{V}[t]$ as a matrix of zeros
4   Initialize $\boldsymbol{C}[t]$ and $\boldsymbol{S}[t]$ based on the best individual
    and global solution
5   Initialize $\boldsymbol{O}[t]$ based on obstacle information
6   Initialize $\boldsymbol{M}[t]$ based on te position of the nearest
    neighbor that has not chosen it
7   Loop:
8     for all robots
9       Evaluate the objective function
10      Calculates the adjacency matrix $\boldsymbol{A}$
11        Update $\boldsymbol{C}[t]$, $\boldsymbol{S}[t]$, $\boldsymbol{O}[t]$ and $\boldsymbol{M}[t]$
12        Calculates $\boldsymbol{V}[t+1]$ and $\boldsymbol{X}[t+1]$
13  end
14  until stopping criteria (convergence)
```

4. Robotic DPSO

The DPSO [3] may be represented by *multiple swarms* of test solutions where each swarm individually performs just like an ordinary PSO algorithm with some rules governing the collection of swarms that are designed to simulate natural selection, like in Darwin's Theory. The selection process implemented is a selection of swarms within a constantly changing collection of swarms. In the common DPSO, 'punish' means deleting particles and swarms, while 'reward' means spawning from new particles to swarms. In order to adapt DPSO to mobile robotics, we propose the deleting and spawning of a robot to be modeled by the mechanisms of *social exclusion* and *social inclusion*, respectively. The following definition of social exclusion was proposed by Burchardt [27]: '*an individual is socially* *excluded if he/she does not participate to a reasonable degree over time in certain activities of his/her society, and* (a) *this is for reasons beyond his/her control, and* (b) *he/she would like to participate*'. In other words, the adaptation of DPSO to MRS, herein denoted as RDPSO, will also be represented by multiple swarms (group of robots) where each swarm individually performs just like RPSO algorithm in search for the solution and some rules governs the whole population of robots.

Algorithm 2: RDPSO Algorithm

Main Program Loop	Evolve Swarm Algorithm
1 For each swarm in the collection	1 For each ***s*** 2 Computes **Algorithm** (RPSO Algorithm)
2 Evolve the swarm ***Evolve Swarm Algorithm*** (**right** →)	3 If swarm ***s*** gets better 4 Reward swarm ***s*** with the best performing robot in the socially excluded group
3 Allow the swarm to spawn a new swarm from the $\boldsymbol{n_1}$ best performing robots in the socially excluded group	5 If swarm ***s*** has not improved
4 Move 'failed' swarms to the socially excluded group	6 Punish swarm ***s*** by excluding the worst performing robot adding it to a socially excluded group

The number of times a swarm s is evolved without finding an improved objective is tracked with a search counter, SC_s. If the swarm's search counter exceeds a maximum critical threshold, $SC^{\max}$, the swarm is punished by excluding the worst performing robot, which is added to a socially excluded group. In this situation, the swarm's search counter is then reset to a value near $SC^{\max}$ that can be calculated using Equation (11).

N_s^{kill} represents the number of robots excluded from the swarm s over a period of time in which there was no improvement in the swarm's objective function.

$$SC_s = SC^{\max}\left[1 - \frac{1}{N_s^{\text{kill}} + 1}\right] \tag{11}$$

The worst performing robot is evaluated by the value of its objective function compared to other members in the same swarm. If the number of robots falls below the minimum acceptable number of robots to form a swarm $n_{\min}$, the swarm is punished by being dismantled and all the robots that belong to that swarm are added to the socially excluded group. On the other hand, if the swarm improves its objective function, then it is rewarded with the best performing robot in the socially excluded group. If a swarm has been more often rewarded than punished, it has a small probability $p = f/S_T$ of spawning a new swarm, where f is a uniform random number on [0,1] and S_T is the number of active swarms. This factor avoids the creation of newer swarms when there are already a large number of swarms. Moreover, the group of robots forming this new swarm will be the best performing robots within the socially excluded group. The key issue in this novel approach is the answer to the question: What do robots belonging to the socially excluded group do? In fact the answer is the same that we would give if asking about a group excluded from our society: they do not do 'anything'. Instead of searching for the objective function's global optimum (i.e. the main activity of the society) like the other robots in the active swarms do, they randomly wander in the scenario. Note, however, that they are always aware of their individual solution and the global solution of the socially excluded group.

Consider the example in Figure 3. Let us suppose a population divided into 3 swarms of 3 robots each (Figure 3(a)). If swarms 1 and 2 (red and green robots, respectively) cannot improve their objective for $SC^{\max}$ iterations they are punished by excluding the worst performing robot of each swarm and adding them to the socially excluded group (Figure 3(a)). The socially excluded robots randomly wander in the scenario memorizing their individual best solution and the global best solution of the socially excluded group (Figure 3(c)). Swarm 3 improves its solution, since it finds a local optimum, and it is rewarded with the best performing robot in the socially excluded group (Figure 3(d)). Finally, the new member of swarm 3 communicates its best individual solution to the other members which is better than their best global solution inducing them to move toward this new solution (Figure 3(e)).

Algorithm 2 summarizes the RDPSO algorithm. This new approach benefits from Darwin's Theory, survival of the fittest, thus presenting a mechanism to escape from local solutions. Furthermore, note that having multiple swarms enables a more distributed approach than in RPSO because the network that was previously defined

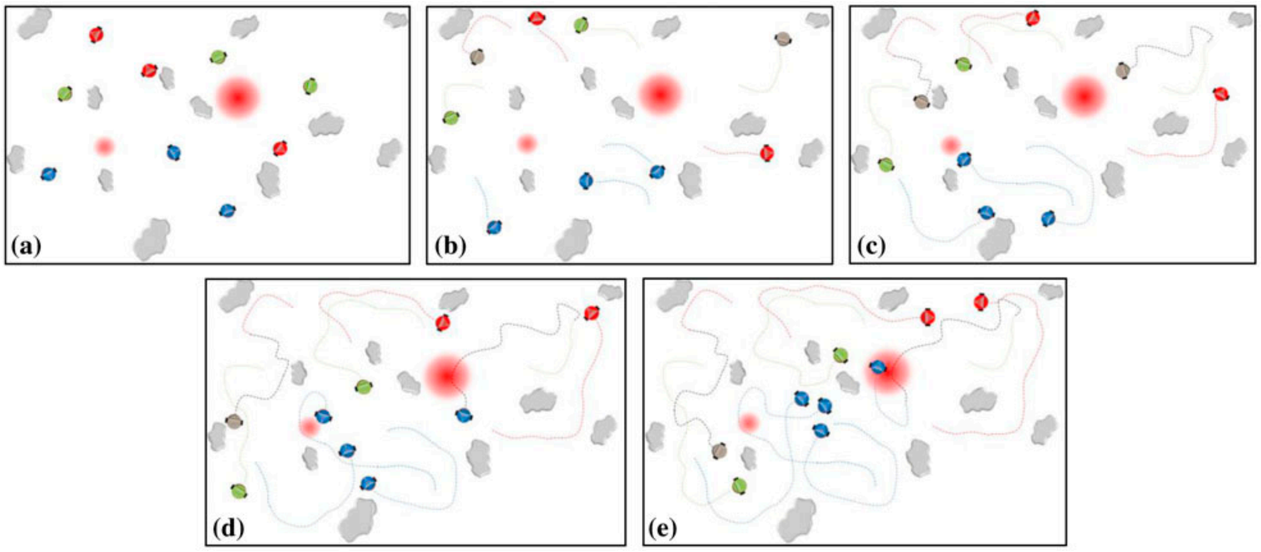

Figure 3. Sequence of an exploration on a scenario with a regular density of obstacles using the RDPSO algorithm.

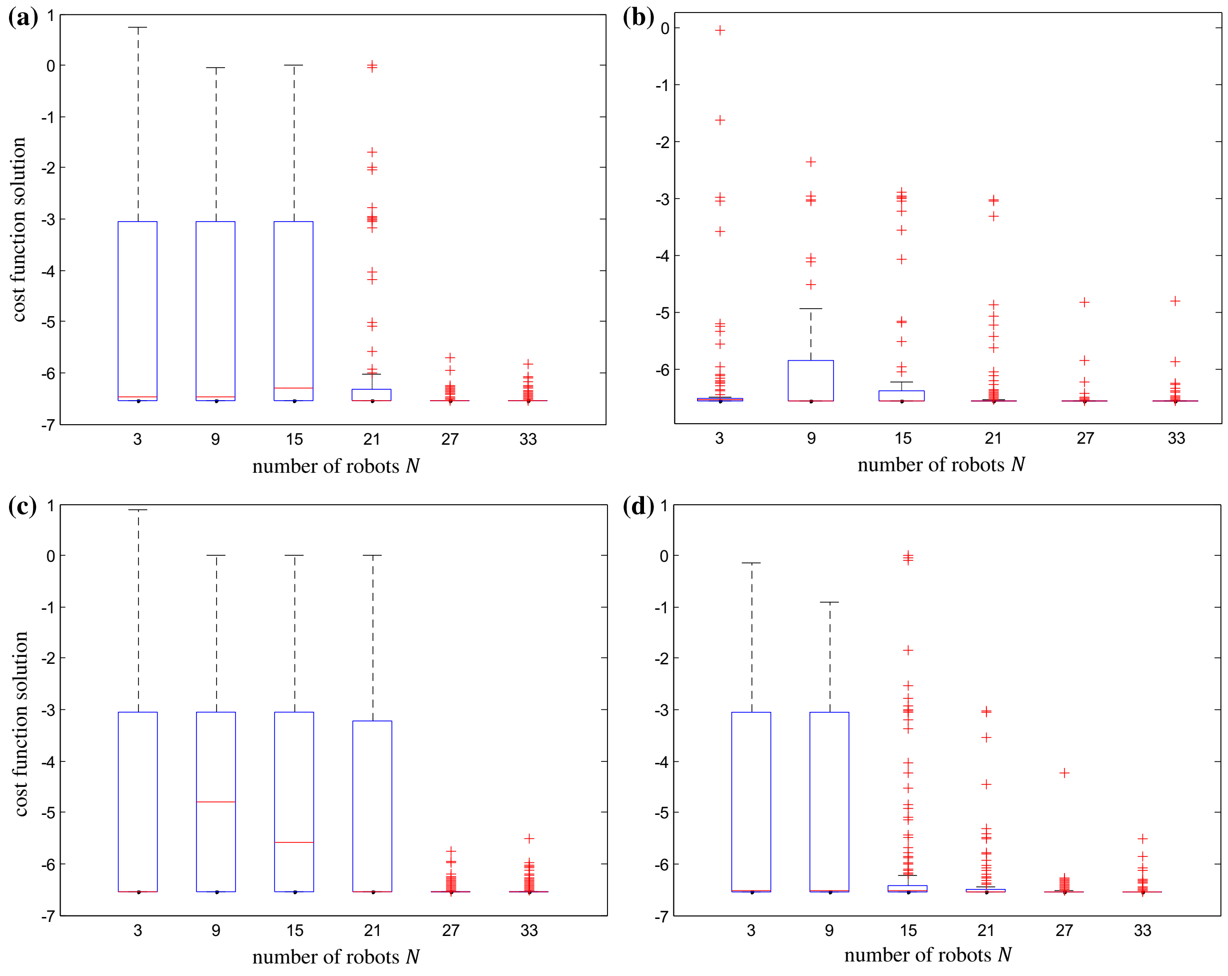

Figure 4. Performance of the algorithms changing the number of robots N in the population; (a) RPSO; (b) RDPSO with a regular density of obstacles; (c) RPSO; (d) RDPSO with a high density of obstacles.

by the whole population of robots is now divided into multiple smaller networks (one for each swarm), thus decreasing the number of nodes (i.e. robots) and the information exchanged between robots of the same network. In other words, robots interaction with other robots through communication is confined to local interactions inside the same swarm, thus making RDPSO scalable to large populations of robots. Furthermore, since swarms are dynamic (members can be punished or rewarded), there is a larger possibility to escape from local solutions when robots are socially excluded or need to travel from one swarm to another. At this point of the work, the way robots travel from their old swarm to a new swarm consists on a simple wandering algorithm until the robot finds its new swarm.

5. Experimental results

In this section, the effectiveness of using a modified version of PSO and DPSO, respectively denoted as RPSO and RDPSO, on a group of robots performing distributed unsupervised learning with local and global information is experimentally assessed with simulations. The number of robots is equal to the number of particles in the population, so each robot is represented by a single unique particle. Robots are randomly deployed in the search space in a spiral manner (cf. Section 3.3) where the radius depends on the maximum communication distance $d_{\max}$ and the number of robots N in the population. For the sake of simplicity and without lack of generality, a distance criterion $d_{\max}$ was used to model communication constraints. Trying to maintain the network connectivity by considering only the communication range $d_{\max}$ does not match reality since the propagation model is more complex – the signal depends not only on the distance but also on the multiple paths from walls and other obstacles. However, the communication distance is a good approach and it is easier to implement. Moreover, the conclusion presented in this section can be extrapolated by replacing maximum distance by minimum signal quality. Experimental results are divided in two stages: (1) evaluate and compare the RPSO and RDPSO algorithms; and (2) understand the relationship between the population of robots N and the maximum communication distance $d_{\max}$ for the best performing algorithm.

5.1. *Comparison of the algorithms*

Multiple test groups of 250 trials with 350 iterations each were considered for both RPSO and RDPSO algorithms. In the particular case of the RDPSO, it is used a minimum, initial and maximum number of 1, 3 and 6 swarms, respectively (represented by different colors in Figure 3), independently of the population of robots taking into account the algorithm description in Section 4, where the number of swarms may vary throughout the simulation. In these experiments, the search space is represented by an example of a Gaussian distribution on a function of two variables of the search space, x and y-axis, which represents the position of the robot in meters. The optimum value of this function (−6.54 in the example) is represented in Figure 5 by a dashed line. The particles will then move in a scenario of size 30×30 m where the z-axis represents the value of the objective function. In this specific case, the objective of the particles is to find the minimum value of the cost. Both algorithms will be evaluated by changing the density of obstacles and the number of robots (i.e. population). Boxplot charts are

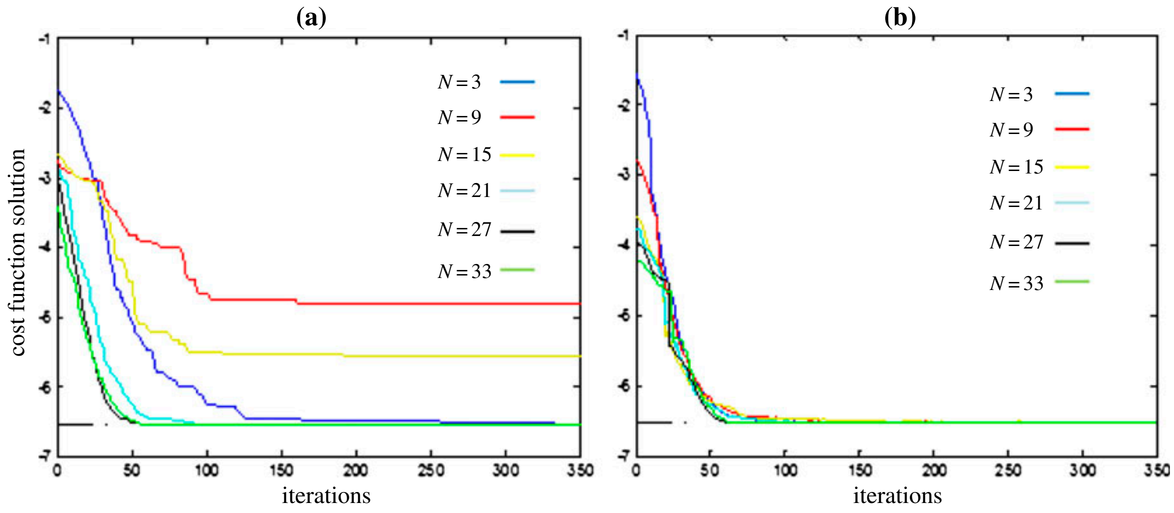

Figure 5. Convergence of the algorithms changing the number of robots N in the population with a high density of obstacles; (a) RPSO; (b) RDPSO.

used to present graphically the final result of each trial. Experiments are then divided into three types: (1) without obstacles; (2) with a regular density of obstacle randomly deployed at each trial; and (3) with a high density of obstacles randomly deployed at each trial.

The experimental results obtained without obstacles are used as guidelines for a better understanding of the impact of obstacles in the algorithm's performance since both algorithms performed efficiently without obstacles and obtained the optimal solution. The number of robots will vary from 3 robots to 33 robots with incremental steps of 6 robots, i.e. $N = \{3, 9, 15, 21, 27, 33\}$ in order to understand the performance of the algorithms related to the population size (Figures 4 and 5). As expected, the rise in the number of obstacles leads to a decrease of performance in both algorithms, for a robot population inferior to 21 robots. It is also clear that the RPSO gets stuck in the local optimum (in the neighborhood of 0 and -3), thus increasing the inconsistency of the final result obtained (larger blue boxes and whiskers). This performance gets better as the number of robots rises. However, there are some situations in which a population of 3 robots performs better than a population of 9 or 15. This happens since dynamic obstacles increases when the population increases. Simulation results show that this drawback can only be overcome using a population superior to 15 robots. It is also verified that for $N \geqslant 27$ the algorithm tends to stabilize and the impact of the presence of obstacles in the algorithm performance is diminished as robots always arrive at the desired destination. The data distribution, despite the considered trial, turns out to be positively skewed (i.e. the mean is higher than the median). This means that, in this case, as the goal is to minimize the cost function, 50% of the trials are around the desired objective value. Nevertheless, the RDPSO shows a better performance when compared with the RPSO in the three experimental datasets, being the median (red line) closer to the objective value, regardless of the number of robots.

Since these simulation experiments represent a search task, it is necessary to evaluate not only the completeness of the mission but also the speed. Therefore, to further compare both algorithms, the convergence of the RPSO and RDPSO can be analyzed for the worst case scenario, i.e. for a high density of obstacles.

As Figure 5 shows, the median of the best solution over the 250 trials was taken as the final output for each value in the set $N = \{3, 9, 15, 21, 27, 33\}$. Once again, the performance of the RDPSO turns out to be better than the performance of the RPSO, with a full convergence to the desired objective value at time $t = 50$, regardless of the number of robots considered. This can be explained by the effect of social exclusion/inclusion described in Section 4 in which the main goal is to avoid being stuck in local optima. As Figure 5(a) confirms, the RPSO algorithm sometimes gets stuck in local solutions.

5.2. Population vs. communication distance

In this section, it is explored the effectiveness of using the RDPSO under different communication constraints and number of robots. Multiple test groups of 100 trials with 300 iterations each were considered. Once again, it is used a minimum, initial and maximum number of 1, 3 and 6 swarms. The search space is represented by a Gaussian distribution consisting on a function of two variables of the search space, x and y-axis, which represents the position of the robot in meters. The robots will then move in a 300×300 m scenario.

In order to improve the interpretation of the algorithm performance, results were normalized in a way that the objective of robotic team is to find the optimal value of 1 while avoiding obstacles and enforcing the MANET connectivity. The maximum communication distance $d_{\max}$ between robots depends on the chosen wireless protocol. Four conditions were described: (1) With communication infrastructure (i.e. without communication constraints $\equiv d_{\max} \to \infty$); (2) *WiFi*; (3) *ZigBee*; and (4) *Bluetooth*. Table 1 depicts the maximum communication distance adapted from the comparison between the key characteristics of each wireless protocol made in [28].

The mean between the minimum and maximum range shown in [28] was considered as the maximum communication distance $d_{\max}$. To demonstrate the performance of the algorithm while constrained by the communication distance, the number of robots is increased until it achieves a mean value equal or superior to 90% of the optimal value 1.

Through the analysis of Figure 6 it can be verified that the number of used robots must be higher in order to obtain a performance closer to the one obtained in a

Table 1. Typical maximum communication distances of the *WiFi*, *ZigBee* and *Bluetooth* protocols.

	WiFi	*ZibBee*	*Bluetooth*
$d_{\max}$ [m]	100	55	10

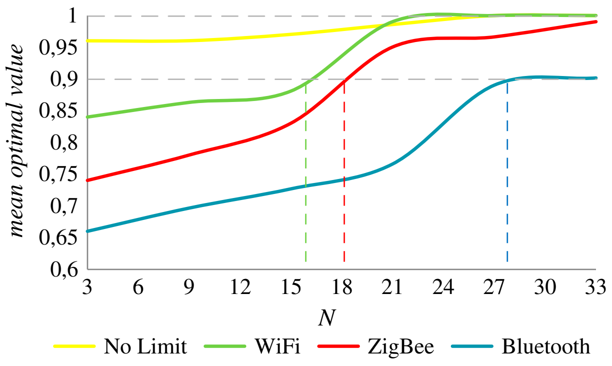

Figure 6. Performance of the algorithm under limited communication distance.

system with no communication constraints. In a 300×300 m scenario, by imposing a 100 m range limitation (i.e. *WiFi*) 16 robots are needed. In the case of *ZigBee*, the range limitation is 55 m, resulting in the need of 18 robots. The number of needed robots dramatically increases to 28 robots when the system has a range limitation of 10 m (i.e. *Bluetooth*). It should be noted that, although *WiFi* and *ZigBee* do not have a significant influence on the performance of the wireless network,[1] the increase in the number of robots (i.e. nodes) from a range of 3 to 33, may have a bigger influence when considering *Bluetooh* as the performance could be slightly decreased since each network (i.e. *piconet*) can only be formed by 8 nodes resulting in the need to interconnect multiple piconets (i.e. *scatternet*).

5.3. Evaluation on real robots

In this section, it is explored the effectiveness of using the RDPSO on swarms of real robots, while performing a collective foraging task with local and global information under communication constraints. Since the RDPSO is a stochastic algorithm, every time it is executed it may lead to different trajectory convergence. Therefore, multiple test groups of 20 trials of 3 min each were considered. A minimum, initial and maximum number of 1, 2 and 3 swarms were used independently of the population of robots.

The *eSwarBot* was the platform used to evaluate the algorithm (Figure 7(a)). It consists on a differential ground platform recently developed and presented in [29] for applications in swarm robotics. Although the platform presents a limited kinematic resolution of 3.6 degrees while rotating and 2.76 mm when moving forward, its low cost and high autonomy allowed to perform experiments with up to 12 robots, with $N = \{4, 8, 12\}$. Robots are equipped with RGB-LEDs on top of them that allow representing a wide range of different colors that matches different swarms. All of the experiments were carried out in an enclosed arena of 2.55–2.45 m that contained two sites (Figure 7(b)). Each site was represented by an illuminated spot uniquely identifiable by controlling the brightness of the light. Despite being an obstacle free scenario, the robots themselves act as dynamic obstacles – note that a maximum number of 12 robots correspond to a population density of approximately 2 robots m^{-2}. Each robot possessed overhead light sensors (LDR) that allowed it to find candidate sites and measure their quality. The brighter site (global optimum) was considered better than the dimmer one (local optimum), and so the goal of the robots was to collectively choose the brighter site. Inter-robot communication to share positions and local solutions were carried out using *ZigBee* 802.15.4 wireless protocol. Since robots were equipped with *XBee* modules that allow a maximum communication range larger than the whole scenario (near 30 m in indoor scenario), robots were provided with a list of their teammates' address in order to simulate the *ad hoc* multihop network communication with limited range. The maximum communication distance between robots $d_{\max}$ was varied between 0.50 and 1.50 m. At each trial, robots were manually deployed on the scenario in a spiral manner while preserving the maximum communication distance $d_{\max}$. The previously described conditions give a total of 120 experiments, thus leading to a runtime of 6 h. The next sequence of frames (Figure 8) presents a trial of the team's performance using $N = 12$ and $d_{\max} = 1.50$ m.

Since these experiments represent a foraging task, it is necessary to evaluate both the completeness of the mission and the time needed to complete it. Therefore, Figure 9 depicts the convergence of the RDPSO for the several proposed conditions. The median of the best solution in the 20 experiments was taken as the final output in the set $N = \{4, 8, 12\}$ for each $d_{\max}$.

Analyzing Figure 9, it is clear that the proposed mission can be accomplished by any number of robots between 4 and 12. In fact, independently on the number of robots, teams converge to the solution in approximately 90% of the experiments. The charts also show that increasing the number of robots slightly increases the convergence time. A population of 4, 8 and 12

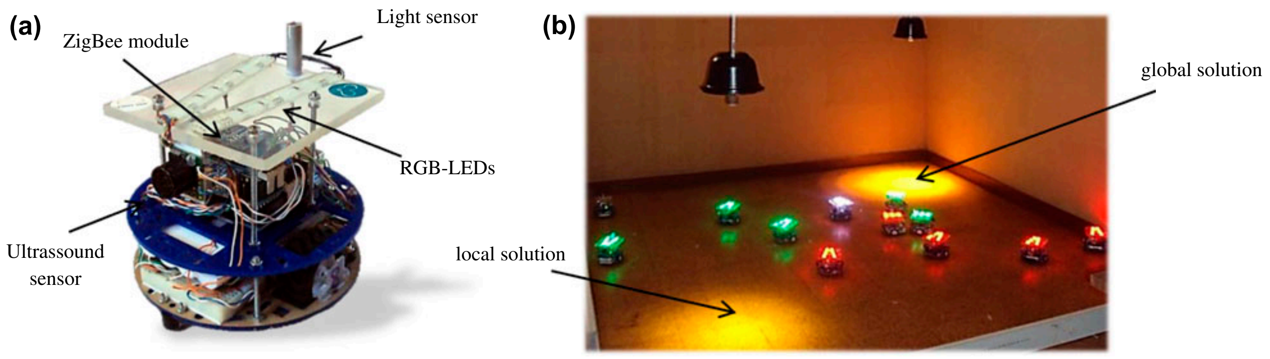

Figure 7. (a) The *eSwarBot* differential ground platform. (b) Experimental setup with 12 *eSwarBot*s.

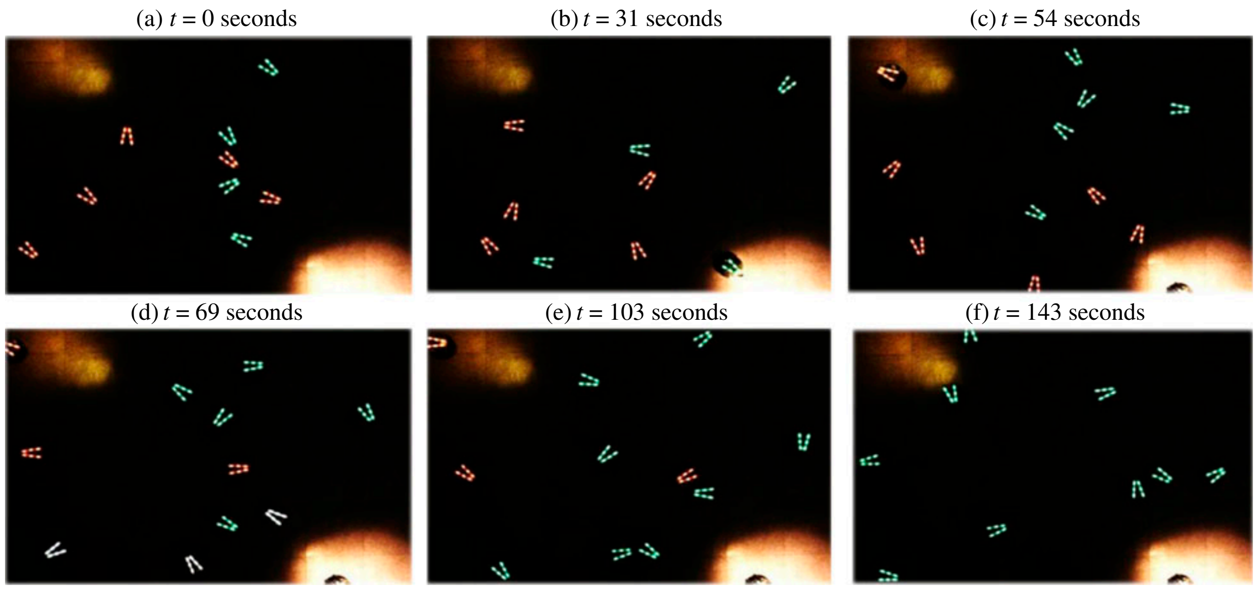

Figure 8. Frame sequence showing the RDPSO performance on a population of 12 robots (some robots may be outside camera's range). (a) The population is initially divided into two swarms – green swarm and red swarm – deployed in a spiral manner; (b) The swarms independently search for the brighter site taking into account a maximum communication distance of 1.5 m between robots of the same swarm; (c) One robot from the red and green swarm finds the local and global optima, respectively; (d) As the red swarm does not improve, some robots are excluded, thus being added to the socially excluded group (white swarm); (e) Since the green swarm has improved, it is able to call new members from the socially excluded group; (f) Finally, the green swarm proliferates calling all the previously excluded robots that were unable to improve their solution. Note that robots do not all converge the global solution as they try to maintain a distance of $d_{\max}$ between them.

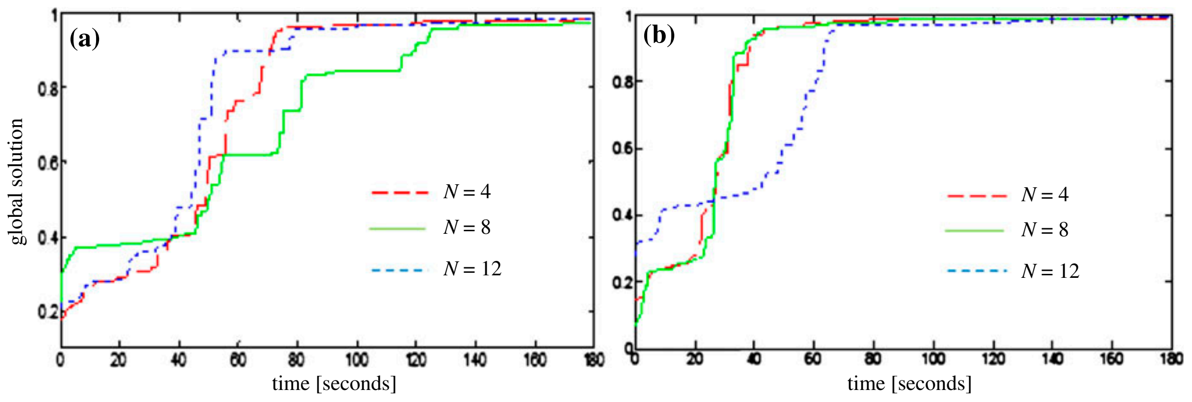

Figure 9. Performance of the algorithms changing the number of robots N in the population: (a) $d_{\max} = 0.5$ m; (b) $d_{\max} = 1.5$ m.

robots takes, in average, 77, 106 and 112 s to converge to the global optimum, respectively. This is a repercussion of having more robots inside the same arena – the number of dynamic obstacles is higher. As expected, increasing the maximum communication distance generally results in a faster convergence to the global solution. Another important factor is that some robots of a given swarm are unable to converge to the final solution when one robot of the same swarm finds it. This issue is related with odometry limitations of the platforms which results in the accumulation of positioning errors. The use of encoders, such as the ones used in these robots, is a classical method, being of low-cost and simple use. However, it is verified the existence of errors inherent to their use are cumulative.

6. Conclusion and future work

Modified versions of the PSO and the Darwinian PSO (DPSO) algorithms based on real-world MRS characteristics such as obstacles avoidance abilities and communication issues were developed and respectively named as RPSO (Robotic PSO) and RDPSO (Robotic DPSO). The features presented in this document were first implemented in a simulated environment and experimental results show how the performance of an MRS with a biologically inspired behaviour based on natural selection and social exclusion, as in the RDPSO, increases when compared to the RPSO. As expected, the influence inherent to communication's limitations can be attenuated as the number of robots or the communication range/ quality increases. This is a promising result for communities of swarm robots with many individuals since they can develop efficient coordination techniques, just like natural swarm agents, allowing cooperative and competitive work in large and super-large societies. Moreover, to further evaluate the RDPSO, experiments with up to 12 physical robots were conducted. Experimental results show that the performance of the robotic population in the proposed scenario is not significantly affected by the number of robots in the population and the maximum communication distance between robots. Nevertheless, one of the main limitations of the algorithm resides in having several parameters, such as the fractional order, influencing the performance of the robotic team. Therefore, one of the future approaches will be the analytical analysis of the RDPSO in order to find a relationship between parameters, thus optimizing the algorithm with regard to the main objective, robot dynamics, obstacles susceptibility, and MANET connectivity.

Acknowledgments

This work was supported by a PhD scholarship (SFRH/BD / 73382/2010) granted to the first author by the Portuguese Foundation for Science and Technology (FCT) and the Institute of Systems and Robotics (ISR) also under regular funding by FCT. This work was made possible by the support and assistance of Carlos Figueiredo and Miguel Luz and for their cooperation, advice and encouragement of this research.

Notes on contributors

Micael S. Couceiro is a Doctoral Candidate on Electrical and Computer Engineering at the Faculty of Sciences and Technology of University of Coimbra. He obtained the MSc degree on Automation and Communications in April 2010, at the Engineering Institute of Coimbra - Portugal. He conducts research on multi-robot systems and swarm robotics at the Mobile Robotics Laboratory from the Institute of Systems and Robotics and at RoboCorp from the Polytechnic Institute of Coimbra. He has published papers on Mobile Robotics, Biomimetics, Fractional-Order Control, Sports Engineering, Biomechanics and Mathematical Methods.

Rui P. Rocha completed his PhD degree in May 2006, at the Faculty of Engineering of the University of Porto. Between February 2000 and May 2006, he was a Teaching Assistant at the Department of Electrical and Computer Engineering, in the Faculty of Sciences and Technology of the University of Coimbra. Currently he is an Assistant Professor at the Department of Electrical and Computer Engineering and a researcher at the Institute of Systems and Robotics, in the Faculty of Sciences and Technology University of Coimbra. His main research topics are cooperative Multi-Robot Systems, 3-D Map Building, Distributed Architectures and Intelligent Transportation Systems.

Nuno M.F. Ferreira graduated with a BSc degree at Engineering Institute of Coimbra in Electrical Engineering (1994). He graduated with a Teaching Licensure (BSc plus 2 years) and MSc at the University of Porto in Electrical Engineering (1999). He obtained the PhD degree in Electrical Engineering at the Faculty of Engineering of the University of Trás-os-Montes e Alto Douro (2006). Since 1997 he works at the Engineering Institute of Coimbra and he is currently a Professor in the Department of Electrical Engineering. His main research topics are Robotics and Control Systems.

Note

1. *WiFi* supports up to 2007 nodes and *ZigBee* theoretically supports up to 65000 nodes.

References

[1] Floreano D, Mattiussi C. Bio-inspired artificial intelligence: theories, methods, and technologies. Cambridge (MA): MIT Press; 2008.

[2] Kennedy J, Eberhart R. A new optimizer using particle swarm theory. Proceedings of the IEEE Sixth International Symposium on Micro Machine and Human Science; 1995; Nagoya, Japan. p. 39–43.

[3] Tillett J, Rao TM, Sahin F, Rao R, Brockport S. Darwinian particle swarm optimization. Proceedings of the 2nd Indian International Conference on Artificial Intelligence; 2005; Pune, India. p. 1474–1487.

[4] Rocha R, Dias J, Carvalho A. Cooperative multi-robot systems: A study of visionbased 3-D mapping using information theory. Robotics and Autonomous Systems 2005;53:282–311.

[5] Sahin CS Urrea E, Hokelek I, Conner M, Uyar M. Self-deployment of mobile agents in MANETs for military applications. Army Science Conference; 2008; Orlando, Florida. p. 1–8.

[6] Bonabeau E, Dorigo M, Theraulaz G. Swarm intelligence: From natural to artificial systems. New York: Oxford University Press; 1999.

[7] Min HQ, Zhu JH, Zheng XJ. Obstacle avoidance. In: Proceedings International Conference on Machine Learning and Cybernetics; 2005; Guangzhou, China. p. 2950–2956.

[8] Pugh J, Martinoli A. Inspiring and modeling multi-robot search with Particle Swarm Optimization. In: Proceedings of the 2007 IEEE Swarm Intelligence Symposium; 2007; Honolulu, Hawaii.

[9] Hereford J, Siebold M. Multi-robot search using a physically-embedded Particle Swarm Optimization. Int. J. Comput. Intell. Res. 2008;4:197–209.

[10] Arrichiello F, Chiaverini S, Fossen TI. Formation control of marine surface vessels using the Null-Space-based Behavioral Control. Group Coord. Cooperative Control. 2006;336:1–19.

[11] Çelikkanat H, Sahin E. Steering self-organized robot flocks through externally guided individuals. Neural Comput. Appl. 2010;19:849–865.

[12] Tardioli D, Mosteo AR, Riazuelo L, Villarroel JL, Montano L. Enforcing network connectivity in robot team missions. Int. J. Rob. Res. 2010;29:460–480.

[13] Tardioli D, Villarroel JL. Real time communications over 802.11: RT-WMP. In: IEEE Internatonal Conference on Mobile Adhoc and Sensor Systems; 2007; Pisa, Italy.

[14] Sheng W, Yang Q, Tan J, Xi N. Distributed multi-robot coordination in area exploration. Rob. Auton. Syst. 2006;54:945–955.

[15] Menezes P. Navegação de Robôs Móveis [Navigation of mobile robots] [MSc Thesis]. University of Coimbra, Coimbra, Portugal; 1999.

[16] Passino KM, Yurkovich S. Fuzzy control. Boston (MA): Addison-Wesley; 1998.

[17] Jatmiko W, Sekiyama K, Fukuda T. Modified Particle Swarm Robotic odor source localization in dynamic environments. Int. J. Intell. Control Syst. 2006;11:176–184.

[18] Williams RL, Wu J. Dynamic obstacle avoidance for an omnidirectional mobile robot. J. Rob. 2012; 2010(no. 901365): 1–14

[19] Miller LE. Multihop connectivity of arbitrary networks. In: Wireless Communication Technologies Group, NIST; 2001; Gaithersburg, Maryland.

[20] Crispin YJ. Cooperative control of multiple swarms of mobile robots with communication constraints. In: Hirsch MJ, Commander CW, Pardalos PM, Murphey R, editors. Optimization and Cooperative Control. Berlin, Heidelberg: Springer-Verlag; 2009. Chapter 381; p. 207–220.

[21] Fink J, Michael N, Kushleyev A, Kumar V. Experimental characterization of radio signal propagation in indoor environments with application to estimation and control. In: IROS'09 Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems; 2009, St. Louis, MO, USA.

[22] Kulkarni RV, Venayagamoorthy GK. Bio-inspired algorithms for autonomous deployment and localization of sensor nodes. IEEE Transactions on Systems, Man and Cybernetics. 2012;40:663–675.

[23] Schwager M, Rus D, Slotine J-J. Unifying geometric, probabilistic, and potential field approaches to multi-robot deployment. Int. J. Rob. Res. 2010;70:21–38.

[24] Akat SB, Gazi V. Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary results. In: IEEE Swarm Intelligence Symposium, St. Louis, MO, USA, 2008.

[25] Tewolde GS, Wu C, Wang Y, Sheng W. Distributed multi-robot work load partition in manufacturing automation. In: 4th IEEE Conference on Automation Science and Engineering, Key Bridge Marriott, Washington DC, USA; 2008.

[26] Hahn HK, Schoenberger K. The ordered distribution of natural numbers on the square root spiral. J. Bus. 2007:1–5.

[27] Burchardt T. Social exclusion: concepts and evidence. In: Breadline Europe: The measurement of poverty; 2000.

[28] Lee JS, Su YW, Shen CC. A comparative study of wireless protocols: bluetooth, UWB, ZigBee, and Wi-Fi. Proceedings of the 33rd Annual Conference of the IEEE Industrial Electronics Society (IECON '07); 2007. p. 46–51.

[29] Couceiro MS, Figueiredo CM, Luz JMA, Ferreira NMF, Rocha RP. A low-cost educational platform for swarm robotics. Int. J. Robots, Educ. Art. 2011;2:1–15.