A fuzzified systematic adjustment of the robotic Darwinian PSO

Micael S. Couceiro, J.A. Tenreiro Machado, Rui P. Rocha, Nuno M.F. Ferreira

ABSTRACT

The Darwinian Particle Swarm Optimization (*DPSO*) is an evolutionary algorithm that extends the Particle Swarm Optimization using natural selection to enhance the ability to escape from sub-optimal solutions. An extension of the *DPSO* to multi-robot applications has been recently proposed and denoted as Robotic Darwinian *PSO* (*RDPSO*), benefiting from the dynamical partitioning of the whole population of robots, hence decreasing the amount of required information exchange among robots. This paper further extends the previously proposed algorithm adapting the behavior of robots based on a set of context-based evaluation metrics. Those metrics are then used as inputs of a fuzzy system so as to systematically adjust the *RDPSO* parameters (*i.e.*, outputs of the fuzzy system), thus improving its convergence rate, susceptibility to obstacles and communication constraints. The adapted *RDPSO* is evaluated in groups of physical robots, being further explored using larger populations of simulated mobile robots within a larger scenario.

Keywords:
Foraging, Swarm robotics, Parameter adjustment, Fuzzy logic, Context-based information, Adaptive behavior

1. Introduction

Mimicking phenomena observed in nature has been the key to the successful development of new approaches in computational sciences (e.g., optimization algorithms [1]) and robotics (e.g., bioinspired robots [2]). Undeniably, the sciences of biomimetics and biomimicry are producing sustainable solutions by emulating nature's time-tested patterns and strategies [1]. Some examples of behavior-based collective architectures, such as ants or bees, inspire the design of novel machine-learning techniques and swarm robotics. This area of research, known as *swarm intelligence* [3,4], studies large collections of relatively simple agents that can collectively solve complex problems. These schemes display the robustness and adaptability to environmental variations revealed by biological agents.

One of the most well-known bioinspired algorithms from swarm intelligence is the Particle Swarm Optimization (*PSO*), which basically consists of a technique loosely inspired by birds flocking in search of food [5]. More specifically, it encompasses a number of particles that collectively move on the search space to find the optimal solution. A problem with the *PSO* algorithm is that of becoming trapped in sub-optimal solutions. Therefore, the *PSO* may work perfectly on one problem but may fail on another. In order to overcome this problem, many authors have suggested extended versions of the *PSO*, such as the Darwinian Particle Swarm Optimization (*DPSO*) [6], to enhance the ability to escape from sub-optimal solutions (cf., [7]). An extension of the *DPSO* to multi-robot applications has been recently proposed and denoted as Robotic Darwinian *PSO* (*RDPSO*), benefiting from the dynamical partitioning of the whole population of robots [8]. Hence, the *RDPSO* allows decreasing the amount of required information exchange among robots and therefore is scalable to large populations of robots [9].

Swarm algorithms such as the *PSO* and its extensions, including the *RDPSO*, present some drawbacks when facing dynamic and complex problems, i.e., problems with many sub-optimal solutions changing over time. The lack of the adaptability to contextual information usually observed in nature turns out to result in sub-optimal solutions that are usually overcome by using exhaustive methods (e.g., sweeping the whole scenario with robots) [10]. For instance, robots in search-and-rescue applications must be efficient in persistently searching for victims while there remains a chance of rescuing them. Although the *RDPSO* previously presented is endowed with punish–reward rules inspired on natural selection to avoid stagnation, robots may take too much time to realize that they are stuck in a sub-optimal solution or that the solution is changing over time. A good example of that may be found on olfactory-based swarming wherein a plume is subject to diffusion and airflow, thus making it hard to find its source (e.g., detection of hazardous gases) [11].

There are two key contributions of this work. First, a set of context-based evaluation metrics, at both the micro- and macro-level, are proposed to assess the *RDPSO* behavior. For that purpose, several concepts inherent to particle swarm techniques (e.g., exploration vs. exploitation) are further studied using two physical platforms with a phase space analysis of their motion (e.g., chaoticity). Secondly, those metrics are used as inputs of a fuzzy system so as to systematically adapt the *RDPSO* parameters (i.e., outputs of the fuzzy system), thus improving its convergence rate, susceptibility to obstacles and communication constraints.

Bearing these ideas in mind, the next section presents some previously developed works to contextualize the approach proposed herein. A brief review of the *RDPSO* algorithm, which benefits from the dynamical partitioning of the whole population of robots into multiple swarms, is given in Section 2. A set of context-based evaluation metrics to measure the collective and individual performance of robots is proposed in Section 3. Subsequently, a novel fuzzy approach to assess the more suitable merging of the evaluation metrics to systematically improve the convergence and performance of the *RDPSO* is presented in Section 4. Populations of real and simulated robots to evaluate the performance of the algorithm are then used in Section 5. Finally, in Section 6 the main conclusions are outlined.

2. Related work

Regardless of *PSO* main variants, the difficulties in setting and adjusting the parameters, as well as in maintaining and improving the search capabilities for higher dimensional problems, is still a matter addressed in recent works [12–14]. Moreover, it is proved that adaptive methods are likely to perform better than nonadaptive methods. For example, one of the most common strategies presented in the literature to solve issues in setting and adjusting *PSO* parameters is based on the stability analysis of the algorithm. In [12], the individual particle's trajectory leading to a generalized model is analyzed, which contains a set of coefficients to control the system's convergence. The resulting system is linear of second-order with stability and parameters depending on the poles, or on the eigenvalues of the state matrix. Kadirkamanathan et al. [13] proposed a stability analysis of a stochastic particle dynamics by representing it as a nonlinear feedback controlled system. The Lyapunov stability method was applied to the particle dynamics in determining sufficient and conservative conditions for asymptotic stability. However, the analysis provided by the authors has addressed only the issue of absolute stability, thus ignoring the optimization toward the optimal solution. More recently, Yasuda et al. [14] presented an activity-based numerical stability analysis method, involved the feedback of swarm activity to control diversification and intensification during the search. The authors showed that the swarm activity can be controlled by employing the stable and unstable regions of *PSO*. However, in a distributed approach such as the *RDPSO*, calculating the swarm activity implies that each robot from the swarm would need to share not only its current position, but also its current velocity with all other members. An alternative to these strategies was accomplished by merging *PSO* algorithms with fuzzy logic. Fuzzy logic was introduced in 1965 by Zadeh [15] at the University of California, Berkeley, to deal with and represent uncertainties. Despite the several possible approaches to implement an online auto-tuning system, fuzzy logic seems to be more adequate to proceed as a multiple criteria analysis tool. The strength of fuzzy logic is that uncertainty can be included into the decision process. Vagueness and imprecision associated with qualitative data can be represented in a logical way using linguistic variables and overlapping membership functions in the uncertain range. For instance, in the work of Shi and Eberhart [16], a fuzzy system is merged into the *PSO* to dynamically adapt the inertia weight of particles. Similarly, Liu et al. [17] presents a fuzzy logic controller to adaptively tune the minimum velocity of the *PSO* particles. Several other authors considered incorporating selection, mutation and crossover, as well as the differential evolution, into the *PSO* algorithm. The main goal is to increase the diversity of the population by either preventing the particles to move too close to each other and collide [18,19] or to self-adapt parameters such as the constriction factor, acceleration constants [20], or inertia weight [21].

Contrary to the multi-robot foraging approach proposed herein, all previously presented works only consider *PSO* and its main variants applied to optimization problems. Robots are designed to act in the real world where both the dynamic and the obstacles need to be taken into account. Furthermore, since in certain environments the communication infrastructure may be damaged or missing (e.g., search and rescue), the self-spreading of autonomous mobile nodes of a mobile ad-hoc network (*MANET*) over a geographical area needs to be considered. Some similar works have been recently presented in the literature. For instance, the work of Saikishan and Prasanna [22] involved the path-planning and coordination of multiple robots in a static-obstacle environment based on the *PSO* and the Bacteria Foraging Algorithm (*BFA*). As the *RDPSO* uses natural selection to avoid getting trapped in sub-optimal solutions, the one proposed by the authors enhances the local search using the *BFA*. Experimental results were conducted in a simulation environment developed in Visual Studio where the pose and shape of obstacles were previously known. However, only one target and two robots were used, thus limiting the evaluation of the proposed algorithm. Hereford and Siebold [23] proposed an embedded version of the *PSO* to swarm platforms. As in the *RDPSO*, there are no central agents to coordinate robots' movements or actions. Despite the potentialities of the physically-embedded *PSO*, the experimental results were carried out using a population of only three robots performing a distributed search in a scenario without sub-optimal solutions. Furthermore, collision avoidance and fulfillment of *MANET* connectivity were not considered.

Despite the accomplishment of other similar works, none of them introduced adaptive behaviors to overcome dynamic properties of real world scenarios. However, the behavior of robots needs to change according to contextual information about the surroundings. This concept of *contextual knowledge* needs to be taken into account to adapt swarms and robots' behavior while considering agent-based, mission-related and environmental context [24]. For example, Calisi et al.'s work [25] presented a context-based architecture to enhance the performance of a robotic system in search and rescue missions using a rule system based on first-order Horn clauses. The set of metrics used as inputs was obtained considering an ''a priori'' map about the difficulty levels concerning mobility and victim detection. Nevertheless, in real applications this would mean a previous knowledge about the scenario, which is not always possible and can be difficult to achieve.

The next section presents the main features of *RDPSO* to help the reader in understanding the introduction to the context-based evaluation metrics subsequently presented.

3. Brief review of the RDPSO

This section briefly presents the *RDPSO* algorithm proposed in [8] and further extended in [9]. Since the *RDPSO* approach is an adaptation of the *DPSO* to real mobile robots, five general features are developed: (i) an improved inertial influence based on

Table 1
Punish–reward *RDPSO* rules.

Punish	Reward
If a swarm does not improve during a specific threshold $SC^{\max}$, then the swarm is punished by excluding the worst performing robot.	If a swarm improves and its current number of robots is inferior to $N_{\max}$, then it is rewarded with the best performing robot that was previously excluded.
If the number of robots in a swarm falls below the minimum number of accepted robots $N_{\min}$ to form a swarm, then the swarm is punished by being dismantled.	If a swarm has been more often rewarded than punished (N_s^{kill} counter), then it has a small probability of spawning a new swarm.

fractional calculus (*FC*) concepts taking into account convergence dynamics; (ii) an obstacle avoidance behavior to avoid collisions; (iii) an algorithm to ensure that the *MANET* remains connected throughout the mission; (iv) a novel methodology to establish the initial planar deployment of robots preserving the connectivity of the *MANET*, while spreading out the robots as most as possible; and (v) a novel punish–reward mechanism to emulate the deletion and creation of robots.

The behavior of robot n can then be described by the following discrete equations at each discrete time, or iteration, $t \in \mathbb{N}_0$:

$$\begin{aligned} v_n[t+1] &= \alpha v_n[t] + \frac{1}{2}\alpha v_n[t-1] + \frac{1}{6}\alpha(1-\alpha)v_n[t-2] \\ &\quad + \frac{1}{24}\alpha(1-\alpha)(2-\alpha)v_n[t-3] \\ &\quad + \sum_{i=1}^{4}\rho_i r_i(\chi_i[t] - x_n[t]), \qquad (1) \end{aligned}$$

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

The parameters α, $0 < \alpha \leq 1$ and ρ_i, $\rho_i > 0$ and $i = 1, 2, 3, 4$, assign weights to the inertial influence, the local best (cognitive component), the global best (social component), the obstacle avoidance component and the enforcing communication component when determining the new velocity. Coefficients r_i, $i = 1, 2, 3, 4$, are random matrices where in each component is generally a uniform random number between 0 and 1. The variables $v_n[t]$ and $x_n[t]$ represent the velocity and position vector of robot n, respectively, and $\chi_i[t]$ denotes the best position of the cognitive, social, obstacle and *MANET* matrix components.

The fractional coefficient α allows describing the dynamic phenomena of the robot's trajectory because of its inherent memory property. The cognitive $\chi_1[t]$ and social components $\chi_2[t]$ are common in the *PSO* algorithm, where $\chi_1[t]$ represents the local best position and $\chi_2[t]$ represents the global best position of robot n. The obstacle avoidance component $\chi_3[t]$ is represented by the position of each robot that optimizes a monotonically decreasing or increasing function $g(x_n[t])$ that describes the distance to a sensed obstacle. In a free-obstacle environment, the obstacle susceptibility weight ρ_3 is set to zero. However, in real-world scenarios, obstacles need to be taken into account and the value of ρ_3 depends on several conditions related with the main objective (i.e., minimize a cost function or maximize a fitness function) and the sensing information (i.e., monotonicity of $g(x_n[t])$). The *MANET* component $\chi_4[t]$ is represented by the position of the nearest neighbor increased by the maximum communication range $d_{\max}$ toward robot's current position. A higher ρ_4 may enhance the ability to maintain the network connected ensuring a specific range or signal quality between robots.

Besides all these components, the *RDPSO* is represented by multiple swarms, i.e., several groups of robots that, together, form the population. Each swarm individually follows Eqs. (1) and (2) in the solution search and some punish–reward rules govern the whole population of robots based on the concept of *social exclusion* (for more details refer to [8]). The *RDPSO* punish–reward rules are summarized in Table 1.

In what concerns the socially excluded robots, instead of searching for the objective function's optimal solution like the other robots in the active swarms, they basically randomly wander in the scenario. This approach improves the algorithm, making it less susceptible of becoming trapped in a sub-optimal solution. However, they are always aware of their individual solution and the global solution of the socially excluded group. Also, having multiple swarms enables a distributed approach, because the network that was previously defined 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 is confined to local interactions inside the same group (swarm), making *RDPSO* scalable to large populations of robots.

In a previous work [26], applying *Jury–Marden's Theorem* [27] to Eqs. (1) and (2) (cf., Appendix A), a convergence analysis of the *RDPSO* was carried out in such a way that the system's convergence is controlled taking into account obstacle avoidance and *MANET* connectivity, without resorting to the definition of any arbitrary or problem-specific parameters. The attraction domain in which the *RDPSO* is stable was then defined by:

$$0 < \alpha \leq 0.632, \qquad (3)$$

$$0 < \sum_{i=1}^{4}\rho_i < 2. \qquad (4)$$

This attraction domain assures the global asymptotic stability of the system (1) and (2) allowing, therefore, robots to find the optimal solution, while avoiding obstacles and ensuring *MANET* connectivity [26]. However, the influence ρ_i and α in the performance of the algorithm needs to be further explored in order to systematically adjust the collective behavior of the swarm.

4. Context-based evaluation metrics

To allow the *RDPSO* adaptive behavior, a set of evaluation metrics, at the macro (i.e., swarm) and micro (i.e., individual robot) levels, that measures the performance of collective movement of mobile robots, needs to be defined. This metrics will be used to systematically adjust the parameters of the algorithm, thus improving its convergence rate, susceptibility to obstacles and communication constraints. Hence, the set of evaluation indices herein proposed are computed, at each iteration, considering environmental and behavioral context. Those measures will then be used as inputs of the fuzzy system in order to control the *RDPSO* parameters (i.e., outputs of the fuzzy system).

To evaluate the following proposed metrics within the *RDPSO* algorithm, a swarm of two physical robots is adopted in the next set of experiments. Robots consisted on differential ground platforms recently developed and presented in [28] for swarm robotics applications denoted as *eSwarBot* s (Educative Swarm Robots). Solutions were defined by illuminated spots on a $2.55\text{ m} \times 2.45\text{ m}$ scenario sensed using the overhead light sensors (*LDR*) of *eSwarBots* (*cf.*, [29] and Section 6 for more detailed description on the experimental setup). Although the platforms present a limited odometric resolution of 3.6° while rotating and 2.76 mm when moving forward, their low cost and high autonomy allow performing experiments with large number of

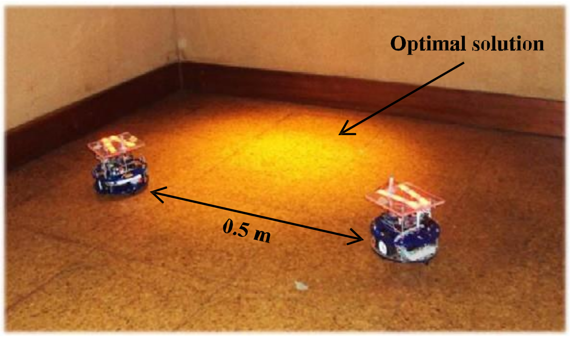


Fig. 1. Experimental setup to evaluate the exploration/exploitation capabilities of a swarm of two robots.

robots. Nevertheless, using only two robots allows easy retrieval of the evolution of each evaluation metric when facing specific extreme situations. For instance, the use of a larger swarm would not drastically affect how one robot behaves when detecting an obstacle within its sensing range. Also, as the scenario has a limited size and number of solutions, a smaller population results in a smaller stochastic effect, thus resulting in negligible differences between different trials despite the existence of the random coefficients r_i, $i = 1, 2, 3, 4$.

4.1. *Exploitation vs. exploration*

As described in [14,30], a swarm behavior can be divided into two activities: (i) exploitation; and (ii) exploration. The first one is related with the convergence of the algorithm, thus allowing a good short-term performance. However, if the exploitation level is too high, then the algorithm may be stuck on sub-optimal solutions. The second one is related with the diversification of the algorithm which allows exploring new solutions, thus improving the long-term performance. However, if the exploration level is too high, then the algorithm may take a long time to find the optimal solution. As first presented by Shi and Eberhart [16], the trade-off between exploitation and exploration in the classical *PSO* has been commonly handled by systematically adjusting the inertia weight. A large inertia weight improves exploration activity while exploitation is improved using a small inertia weight.

Since the *RDPSO* presents a fractional calculus (*FC*) strategy to control the convergence of the robotic team, the coefficient α needs to be systematically adjusted in order to provide a high level of exploration while ensuring the optimal solution of the mission. In order to understand the relation between the fractional coefficient α and the *RDPSO* exploitation/exploration capabilities, the center-of-mass trajectory in phase space of a swarm of two physical robots, for various values of α, while fixing $\rho_i = 0.5$, will be analyzed. Both robots were randomly placed in the vicinity of the solution in (0, 0) with a fixed distance of 0.5 m between them (Fig. 1).

As it may be perceived (Fig. 2), the swarm behavior is susceptible to variations in the value of α. Fig. 2 depicts that when α is too small, i.e., $\alpha = 0.010$, the exploitation level is too high, being likely to get stuck in a sub-optimal solution. However, the intensification of the algorithm convergence is improved—it presents a quick, almost linear, convergence. When α is at the boundary of the attraction domain (cf., Section 3), i.e., $\alpha = 0.632$, the trajectory of the swarm is cyclical and presents a good balance between exploitation and exploration. In this case, robots exhibit a level of diversification adequate to avoid sub-optimal solutions and a considerable level of intensification to converge to the optimal solution, i.e., it presents a spiral convergence toward a nontrivial attractor. When α is too high and outside the attraction domain, i.e., $\alpha = 0.990$,

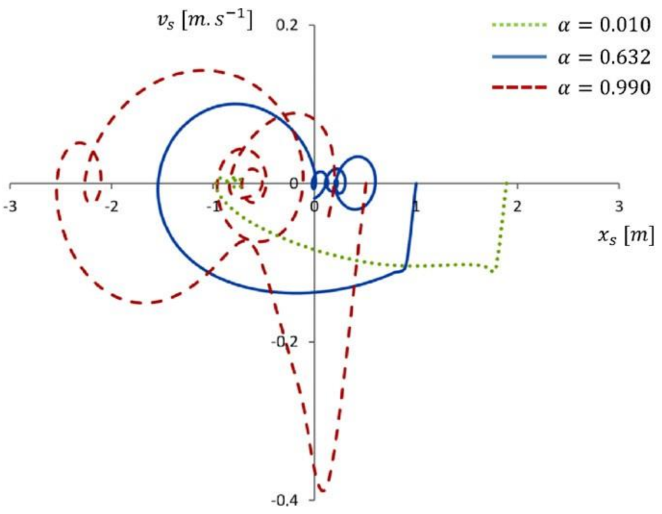


Fig. 2. Center-of-mass trajectories in phase space of a swarm of 2 robots.

despite the cyclical trajectory of the swarm toward the optimal solution, the swarm presents an oscillatory behavior. This results in a high exploration level being more unstable and sometimes unable to converge, i.e., it presents a difficult convergence.

We observe that α needs to be adjusted depending on the contextual knowledge for behavior specialization. Hence, the introspective knowledge about the swarm activity is used to obtain smooth transitions between behaviors. However, a method to evaluate the current swarm activity needs to be considered.

As previously described, the swarm activity in [14] is controlled by switching between the stable and unstable regions of the *PSO*. In our situation, the stable region is defined by the attraction domain presented in Section 3 and previously introduced in [26], wherein the swarm activity is, predominantly, of exploitation. Since the equilibrium between exploitation and exploration is at the boundary of the attraction domain ($\alpha = 0.632$), α should always converge to this value.

Contrarily to [14], in which the activity is defined as the root mean square velocity of particles, let us define the *swarm activity* of swarm s as the norm of the velocity of its center-of-mass $v_s[t]$ at each iteration, i.e., group velocity:

$$A_s[t] = \frac{\|v_s[t]\|}{v_{\max}}, \tag{5}$$

wherein threshold $v_{\max}$ corresponds to the maximum step between iterations. The redefinition of swarm activity was considered in order to underline the collective activity (at the macro level) instead of the sum of activities performed by each robot. Considering the definition presented in [14], robots may present a high activity but the swarm as a whole may present a small activity, i.e., $\overrightarrow{v_s[t]} \approx 0$. Therefore, a swarm activity of $A_s[t] = 0$ means no swarm activity at all and α should increase, while $A_s[t] = 1$ corresponds to a highly chaotic behavior and α should decrease.

It should be noted that this adapted behavior occurs at the collective level. However, the individual behavior of each robot also needs to be considered. By other words, the same swarm may have both exploring and exploiting robots and that state will depend on their cognition and socialization level.

4.2. *Cognition vs. socialization*

Despite the relation between the fractional coefficient α and swarms behavior, it is the combination of all *RDPSO* parameters that determines its convergence properties. The values of both

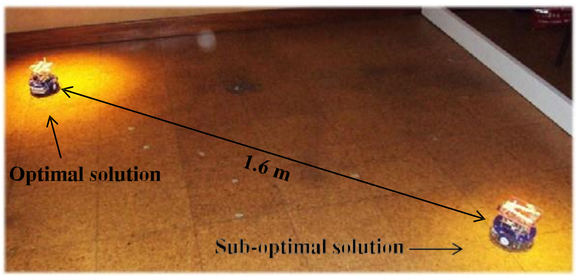


Fig. 3. Experimental setup to evaluate the cognition/socialization between two robots of the same swarm.

cognitive and social factors ρ_1 and ρ_2 are not critical for the algorithm, but selection of proper values may result in better performance, both in terms of speed of convergence and alleviation of sub-optimal solutions. Furthermore, their values have to be taken into account when choosing the fractional coefficient α.

The cognitive component ρ_1 represents the personal ''thinking'' of each robot, thus encouraging robots to move toward their own best positions found so far. The social component ρ_2 represents the collaborative effect of the swarm in finding the optimal solution, thus summoning robots toward the global best position found so far. Venter's work [31] presented experimental results in which a small cognitive coefficient ρ_1 and large social coefficient ρ_2 could significantly improve the performance of the algorithm. However, it should be highlighted that, for problems with multiple sub-optimal solutions, a larger social coefficient ρ_2 may prematurely mislead all robots toward a sub-optimal solution in which they will be unable to avoid since they are ''blind'' followers. On the other hand, a larger cognitive coefficient ρ_1 may cause each robot to be attracted to its own personal best position to a very high extent, resulting in excessive wandering.

To further understand the cognitive and social components of the *RDPSO*, let us then consider an experimental setup of a swarm of two robots. Each robot is initially placed near the sub-optimal and optimal solutions uniquely identifiable by controlling the brightness of the light. The brighter site (optimal solution) is considered better than the dimmer one (sub-optimal solution), and so the goal of the swarm is to collectively choose the brighter site (Fig. 3). It is noteworthy that using a large population of robots within such scenario would not yield much different results as the swarm global best would be collectively chosen as the same than using two robots. In other words, increasing the number of robots would not only increase the variability of the behavior before the collective agreement on the global best solution, as it would significantly increase the complexity on analyzing the evolution of the group.

At the beginning, robots are at a distance of 1.6 m from each other. Also, the fractional coefficient α is now fixed at 0.632 (threshold stability) and $\rho_3 = \rho_4 = 0.1$ for multiple (ρ_1, ρ_2) combinations while keeping the same absolute value $\rho_T = 1$ with $\rho_T = \rho_1 + \rho_2$.

Fig. 4 presents the Euclidean distance in phase space between the two robots, thus depicting the evolution and convergence of the distance between them. Note that the inter-robot distance turns out to represent the distance between the robot located in the sub-optimal solution and the location of the optimal solution itself. This phenomenon can be explained by how the other parameters are defined (more specifically the smaller values of ρ_3 and ρ_4) and the nonexistence of any other better solution within the swarm. Hence, the decision of the robot located in the optimal solution is not disturbed, thus staying still until a better solution is found (which never happens in such a situation).

As expected, increasing the social weight ρ_2 decreases the Euclidean distance between robots, i.e., the distance between

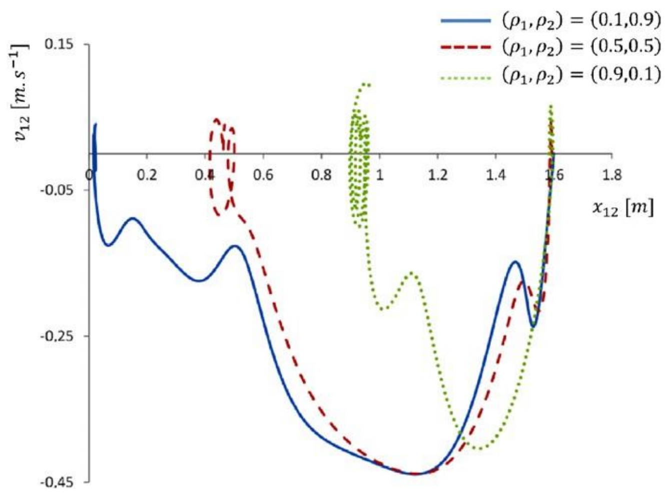


Fig. 4. Distance between robots in phase space to evaluate the relation between ρ_1 and ρ_2.

robots tended to only a few centimeters when using $(\rho_1, \rho_2) = (0.1, 0.9)$ and near 1 m using $(\rho_1, \rho_2) = (0.9, 0.1)$. However, the relation between the final inter-robot distance and (ρ_1, ρ_2) weights is not linear. It can also be observed that, increasing the social weight ρ_2, the robot initially located at the sub-optimal solution converges in a more intensive way, that is, the radius of the spiral at robot's convergence position is smaller for higher ρ_2 values. Hence, the exploitation behavior increases as the distance between robots decreases, thus compromising the performance of the swarm. Moreover, robots' velocity does not directly depends on the relation between ρ_1 and ρ_2, since the relative velocity between robots reaches a maximum velocity of approximately $0.45\ \mathrm{m\ s^{-1}}$ in the three (ρ_1, ρ_2) combinations.

A balance between cognitive and social weights needs to be established and adapted throughout the mission depending on contextual mission-related knowledge of the cognitive or social levels of robots, thus resulting in a different social weight ρ_2 (and, therefore, cognitive weight ρ_1), for each robot.

Suresh et al.'s work [32] presented an inertia adaptive *PSO* in which the modification involved the modulation of the inertia factor according to distance of particles of a particular generation from the global best. Similarly, a micro level metric, defined as *robot socialization*, is then defined as the current Euclidean distance of robot n from its swarm global best:

$$S_n[t] = 1 - \frac{\|\chi_2[t] - x_n[t]\|}{\|\chi_2[t] - \max x_n[t]\|}. \qquad (6)$$

The social level of a given robot will then be the relation between its distance to the global best and the distance to the global best of the farthest robot of the same swarm. Therefore, a robot with a social level of $S_n[t] = 0$ means that it is the farthest robot of the swarm to the best robot and ρ_2 should increase, thus decreasing ρ_1. On the other hand, as robot social level increases, i.e., the distance of a robot to the optimal solution decreases, ρ_2 should decrease (increasing ρ_1). This modulation ensures that in case of robots that have moved away from the global best, the effect of attraction towards the global best will predominate.

Depending on the social level, the fractional coefficient α should vary. As $S_n[t]$ decreases, α should also decrease so that whenever a robot moves far away from the globally best position found so far by the swarm, the effect of its inertial velocity will be minimal. The opposite situation can also be considered. As $S_n[t]$ increases, i.e., the robot gets closer to the global best position, α should increase to present a higher diversification level, thus increasing the possibility to find an improved or alternative solution. Consequently, there may also be a different α for each robot depending on its social level.

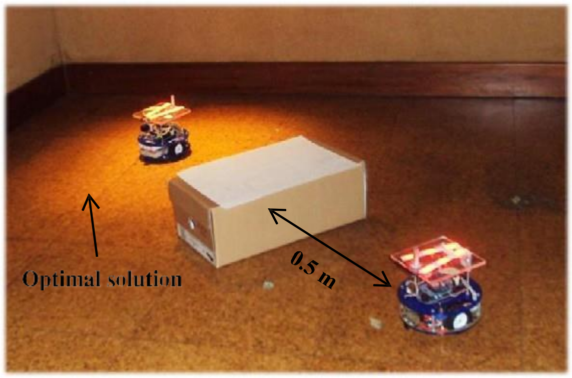


Fig. 5. Experimental setup to evaluate the obstacle susceptibility of a robot.

4.3. Obstacles susceptibility

Using multiple mobile robots for hazardous target search applications requires an efficient way for avoiding obstacles while completing their main mission. The presence, or absence, of obstacles can affect the efficiency of the *RDPSO* since one set of parameters may result in fast convergence but fail in the presence of obstacles or it may increase obstacles susceptibility but swarms may be more resilient.

As previously explained, a robot is able to avoid obstacles due to a repulsive force based on a monotonic and positive *sensing function* $g(x_n)$ $g(x_n[t])$ that depends on the distance between the robot and the obstacle [8]. Its susceptibility is defined through the obstacle susceptibility weight ρ_3. Since the characteristics of the environment are generally not known in advance, the robot itself should be able to intelligently change its own obstacle susceptibility ρ_3 based on the contextual information about the environment.

By means of Eq. (1) one can perceive that, when a robot does not sense any obstacle within its sensing radius r_s, the position that optimizes the monotonically decreasing or increasing sensing function $g(x_n[t])$ is the same as the robot's current position, i.e., $x_n[t] = \chi_3[t]$. This yields the following expression:

$$\rho_3 r_3(\chi_3[t] - x_n[t]) = 0. \tag{7}$$

One may consider that, when a robot does not sense any obstacles within r_s, then the obstacle coefficient should be ignored, i.e., $\rho_3 = 0$. Also, it is easy to remark that its obstacle susceptibility weight should increase as the distance to the obstacle decreases. However, as previously highlighted, it is not the absolute value of a coefficient that matters but the relation between all coefficients. Therefore, for better understanding the relation between ρ_3 and the rest of the *RDPSO* parameters, let us consider a new experimental setup of a swarm of two robots. One of the robots is placed in the optimal solution (i.e., the brighter site), and will summon the other robot towards it. The other robot is placed 1 m away from the best performing robot and an obstacle is placed halfway the path between both robots, i.e., 0.5 m in front of the robot that is being summon (Fig. 5). Also, robots are programmed to detect obstacles at 0.5 m from them, i.e., $r_s = 0.5$.

To allow the manipulation of ρ_3 within a considerable range, while respecting the attraction domain represented by condition (4), let us suppose the following set of parameters $\rho_4 = 0.1$ and $\rho_T = 0.7$, with $(\rho_1, \rho_2) = (0.2, 0.5)$. In other words, the social component influences more than the cognitive one, thus allowing for the robot placed in the optimal solution to promptly lure the other one. As Fig. 6 depicts, the obstacle susceptibility of the robot was evaluated using the following parameters $a = 0.632$ and $\rho_2 = \{0.4, 0.8, 1.2\}$.

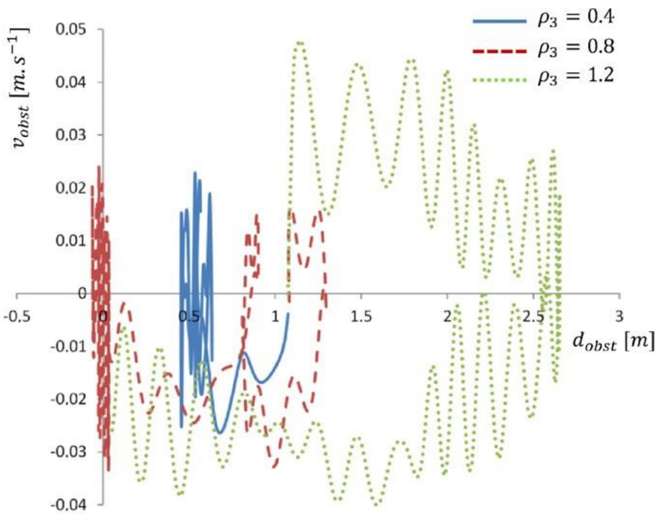


Fig. 6. Distance from the worst performing robot to the obstacle in phase space.

Observing Fig. 6, we conclude that the worst performing robot gets stuck in the obstacle vicinities, and sometimes collides with them, for an obstacle susceptibility weight of $\rho_3 = 0.4$. For any of the other two situations ($\rho_3 = 0.8$ and $\rho_3 = 1.2$), the robot is able to circumvent the obstacle, thus reaching the optimal solution. However, notwithstanding the same final result for both $\rho_3 = 0.8$ and $\rho_3 = 1.2$, as ρ_3 increases the robot presents a more chaotic behavior, i.e., more oscillatory. For $\rho_3 = 1.2$ the robot first moves 1 m and a half away from its current location avoiding the obstacle in an inadequate way.

In fact, as a robot avoids an obstacle, ρ_3 should decrease allowing a wider range of possibilities for the other coefficients, such as ρ_1 and ρ_2. For that reason, the following environmental contextual information about *robot avoidance* was defined:

$$O_n[t] = \frac{r_s - g(x_n[t])}{r_s}, \tag{8}$$

wherein the monotonic and positive *sensing function* $(x_n)g(x_n[t])$ returns r_s when the robot does not sense any obstacle within its sensing radius. As Eq. (8) shows, as an obstacle enters a robot's sensing radius, $O_n[t]$ tends to 1, thus presenting the proximity to the obstacle. On the other hand, when $O_n[t] = 0$, i.e., the robot is in an obstacle-free path, then the obstacle susceptibility weight can be neglected, i.e., $\rho_3 = 0$. However, as $O_n[t]$ increases, the obstacle susceptibility weight ρ_3 should also increase, thus decreasing ρ_T in order to respect the attraction domain defined by conditions (3) and (4).

4.4. Connectivity susceptibility

Wireless networks play a crucial role in *MRS* since robots need to share information to infer their individual locations and solutions, and control their position and orientation to maintain network connectivity. The requirement to ensure network connectivity often fails when robots move apart from their teammates.

To improve the convergence rate of the *RDPSO* robots within the same swarm, robots should spread out as much as possible. However, they must keep a maximum communication distance, or minimum signal quality, between them. In this perspective, one needs to find a good compromise between the enforcing communication component ρ_4 and the mission parameters (i.e., ρ_1 and ρ_2) since each robot has to plan its moves while maintaining the *MANET* connectivity.

The *RDPSO* takes use of the adjacency matrix A that directly depends of link matrix $L = \{l_{ij}\}$ to identify the minimum/maximum

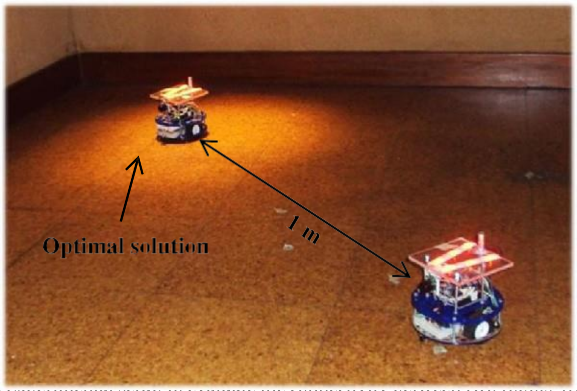


Fig. 7. Experimental setup to evaluate the connectivity between two robots from the same swarm.

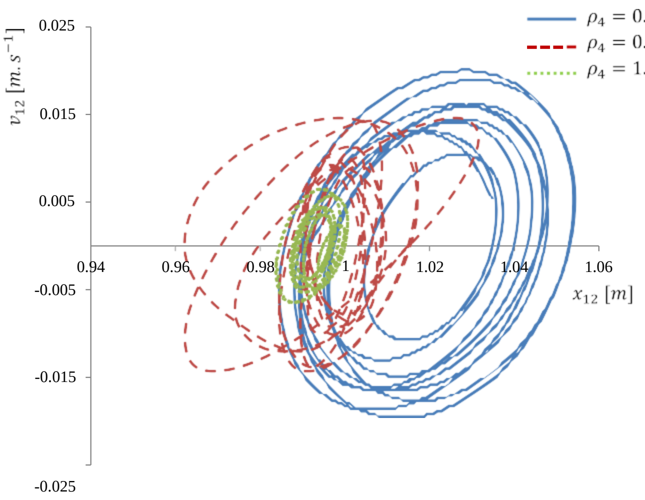


Fig. 8. Distance between robots in phase space.

distance/signal quality of each line, thus returning the position of the nearest neighbor in which a robot needs to ensure connectivity (cf., [9]). Similar to the above methodology, let us now consider an experimental setup of a swarm of two robots. Once again, one of the robots is placed in the optimal solution while the other robot is located 0.5 m away from it (Fig. 7).

The distance between robots x_{12} will be evaluated manipulating ρ_4 within a larger range while respecting the attraction domain represented by condition (4), $\rho_3 = 0.1$ and $\rho_T = 0.7$ with $(\rho_1, \rho_2) = (0.2, 0.5)$, and $a = 0.632$. The enforcing communication component ρ_4 will be set as $\rho_4 = \{0.4, 0.8, 1.2\}$ and robots will try to maintain a distance of 1 m between them, i.e., $d_{\max} = 1$ m (Fig. 8).

It may be observed in Fig. 8 that, for any value of ρ_4, the robot presents a spiral convergence in $d_{\max}$ vicinities. However, as ρ_4 increases, the convergence of the robot toward $d_{\max}$ also increases (the center of the spiral approximates $d_{\max}$). For $\rho_4 = 0.4$, the robot converges towards a distance superior to $d_{\max}$ with a larger spiral radius since it tries to get closer to the solution. For $\rho_4 = 1.2$, the robot ignores the solution and hardly moves from its initial position.

Considering the previous results, the easiest way to ensure connectivity is to increase the enforcing communication component ρ_4 when the distance between robots approximates the threshold value (i.e., maximum distance or minimum signal quality). Therefore, exploiting introspective knowledge allows defining an agent-based contextual metric denoted as *robot proximity*. Nevertheless, this metric will depend on either ensuring a maximum communication distance between robots, $d_{\max}$, or getting a minimum signal quality, $q_{\min}$. In real situations, fulfilling the network connectivity by only taking into account 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, in simulation, the communication distance may be a good approach and it is easier to implement.

Considering a $d_{\max}$ problem, one can define robot proximity as follows:

$$P_n[t] = \begin{cases} 1 - \frac{d_{nm}[t]}{d_{\max}}, & d_{nm}[t] \leq d_{\max} \\ 0, & d_{nm}[t] > d_{\max}, \end{cases} \tag{9a}$$

where $d_{nm}[t]$ is the distance between robot n and its nearest neighbor m. Similarly, considering a $q_{\min}$ problem, the metric will be defined as:

$$P_n[t] = \begin{cases} 1 - \frac{q_{\min}}{q_{nm}[t]}, & q_{nm}[t] \geq q_{\min} \\ 0, & q_{nm}[t] < d_{\min}, \end{cases} \tag{9b}$$

where $q_{nm}[t]$ is the minimum signal quality between robot n and its nearest neighbor m.

Using only inter-robot relations allows ensuring the *MANET* connectivity only locally. Therefore, besides the proposed micro level metric, a macro level metric needs to be defined to globally improve the *MANET* fault-tolerance within each swarm. As presented in Nathan et al.'s work [33], the connectivity of the network can be represented by the second smallest eigenvalue, also known as the *Fiedler value*, λ_2 of the *Laplacian matrix* L defined by:

$$L = \Delta - A, \tag{10}$$

wherein Δ is the *valency matrix* (i.e., *diagonal matrix*).

$$\Delta = \mathrm{diag}\left(\sum_{j=1}^{n} a_{ij}\right). \tag{11}$$

It is noteworthy that the graph is connected when the Fiedler eigenvalue is greater than zero, i.e., $\lambda_2 > 0$. Also, the value of λ_2, depending on the number of robots within a swarm, allows evaluating its connectivity. Therefore, a new macro agent-based contextual metric that takes into account the *swarm connectivity* can be defined as:

$$C_s[t] = \begin{cases} \frac{\lambda_2}{N_S}, & \lambda_2 \geq 0 \\ 0, & \lambda_2 < 0. \end{cases} \tag{12}$$

When all robots within a swarm can directly communicate (i.e., one hop) with all their teammates, then $\lambda_2 = N_S$, thus resulting in $C_s[t] = 1$ which is representative of a fully connected swarm. Therefore, as $C_s[t]$ tends to 0, ρ_4 should increase in order to ensure a more connected *MANET*.

4.5. Summary

The above presented context-based metrics can be used as benchmark to evaluate the *RDPSO* in terms of group behavior. However, the fact that there are multiple evaluation metricsto determine the algorithm's performance makes their selection process complex. Due to the *RDPSO* dynamics, it may not be sufficient to consider each evaluation metric independently. It is thus extremely important to find a way to evaluate its performance and ponder, simultaneously, the full set of metrics.

In this line of thought, it is based on the fuzzy approach, introduced in next section, that we will evaluate the performance and adaptively adjust the parameters of the *RDPSO*.

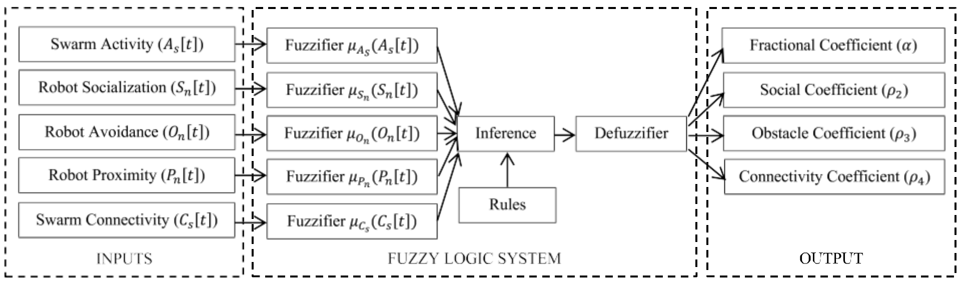

Swarm Activity ($A_s[t]$)
Robot Socialization ($S_n[t]$)
Robot Avoidance ($O_n[t]$)
Robot Proximity ($P_n[t]$)
Swarm Connectivity ($C_s[t]$)
INPUTS
Fuzzifier $\mu_{A_s}(A_s[t])$
Fuzzifier $\mu_{S_n}(S_n[t])$
Fuzzifier $\mu_{O_n}(O_n[t])$
Fuzzifier $\mu_{P_n}(P_n[t])$
Fuzzifier $\mu_{C_s}(C_s[t])$
Inference
Defuzzifier
Rules
FUZZY LOGIC SYSTEM
Fractional Coefficient (α)
Social Coefficient (ρ_2)
Obstacle Coefficient (ρ_3)
Connectivity Coefficient (ρ_4)
OUTPUT

Fig. 9. Fuzzy logic system to control the behavior of the *RDPSO*.

5. Fuzzified systematic parameter adjustment

Robots' perception can significantly benefit from the use of contextual knowledge. The previous section presented the acquisition of environmental knowledge based on the sensing capabilities and shared information between teammates. Fuzzy logic will now be incorporated into the *RDPSO* algorithm to handle contextual information represented by the previously defined metrics. Other proposals with different formalisms to represent contextual knowledge and reason such as Bayesian decision analysis could be adopted as well [24,34]. Nevertheless, fuzzy logic addresses such applications perfectly as it resembles human decision making with an ability to generate precise solutions from certain or approximate information. The successful development of a fuzzy model is a complex multi-step process, in which the designer is faced with a large number of alternative implementation strategies and attributes [35]. In sum, based on the information extracted from the inputs represented by the previously defined metrics, the fuzzy logic system can infer contextual knowledge which can be used to control the *RDPSO* behavior by adapting its parameters (Fig. 9).

This control architecture is executed at each iteration t, thus returning the fractional, social, obstacle and connectivity coefficients, α and ρ_i, $i = 2, 3, 4$. Subsequently, the cognitive coefficient ρ_1 is then defined in order to respect condition (4), i.e., $0 < \sum_{i=1}^{4} \rho_i < 2,\ i = 1, 2, 3, 4.$

As Fig. 9 depicts, the overall organization of this architecture resembles the commonly used feedback controllers wherein contextual knowledge is extracted from data followed by a reasoning phase to control the robot. Hence, based on the metrics previously presented and their definition, one can assess the relation between the inputs and outputs of the fuzzy system. To soften the decision-making system, the membership functions will be defined by generalized bell-shaped functions. The generalized bell-shaped function has one more parameter than the typical Gaussian function used in membership functions being defined as:

$$\mu_X(X[t]) = \frac{1}{\left[1 + \frac{(X[t]-c)}{a}\right]^{2b}} \qquad (13)$$

where parameters a, b and c correspond to the width, the slope and the center of the curve, respectively. Since metrics are all defined between 0 and 1, only half a curve is required to represent the status of the swarm and robots, i.e., $c = 1$. On the other hand, for a soften response, the width and slope may be defined as $a = 0.5$ and $b = 3$ (Fig. 10).

The swarm activity membership function $\mu_{A_S}(A_s[t])$ represents how *Active* the swarm is. As for the robot socialization $\mu_{S_n}(S_n[t])$, it represents how *Social* a robot is. The same analysis can be made for the obstacle avoidance membership function $\mu_{O_n}(O_n[t])$, thus representing how *Close* a given robot is to obstacles. The robot proximity membership function $\mu_{P_n}(P_n[t])$ represents how *Far* a robot is from its neighbor. Finally, as for the swarm connectivity membership function $\mu_{C_s}(C_s[t])$, it was defined to represent how *Connected* the swarm is.

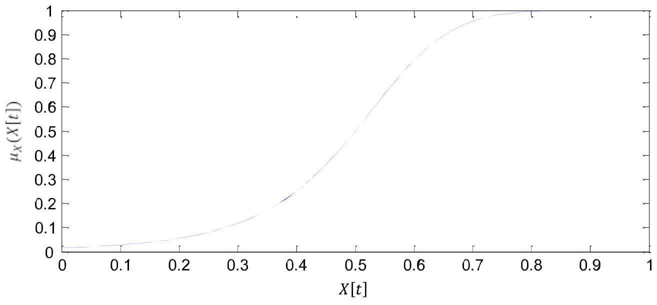


Fig. 10. General membership function for each input.

As for the consequent functions, based on the preliminary experimental assessments presented in the previous section, one can define the following triangular membership relations represented in Fig. 11. These functions not only allow softening and verbalizing the outputs, but also and more importantly, normalizing them within the attraction domain presented in [26]. It is noteworthy that, as previously mentioned (cf., Section 3), the cognitive parameter can then be defined as $\rho_1 = 2 - \rho_2 - \rho_3 - \rho_4$.

The *Mamdani-Minimum* was used to quantify the premise and implication. The defuzzification was performed using the center-of-gravity (*CoG*) method. The *CoG* is a continuous method and one of the most frequently used in control engineering and process modeling being represented by the centroid of the composite area of the output fuzzy term. By using the contextual knowledge represented in Fig. 11, one can define the contextual rules that affect the behavior of the system depending on the situation at hand. Therefore, the following diffuse *IF-THEN-ELSE* rules (cf., [36]) are considered (see Fig. 12):

The rules turn out to prioritize some *RDPSO* parameters over others, in which ρ_3 (i.e., avoiding obstacles) and ρ_4 (i.e., maintaining *MANET* connectivity), are the most pertinent parameters. Although minor collisions are acceptable in swarm robotics, as this work focus on realistic applications such as search-and-rescue, it may be debatable to prioritize the mission over obstacle avoidance. The loss of multiple robots may jeopardize the mission objective (e.g., find victims). On the other hand, it is noteworthy that the obstacle avoidance parameter ρ_3 only affects the behavior of a specific robot when an obstacle is in its sensing range.

In brief, the fuzzy system proposed herein systematically adjusts the behavior of the *RDPSO* in such a way that one can easily understand the contextual information regarding the robot and the swarm by simply observing the parameters' evolution. Hence, the use of contextual knowledge improves robots' perception by allowing fast detection of environmental or mission changes (e.g., detecting an obstacle) exploiting the information about the dynamics of real-world features. For example, Fig. 13 depicts the evolution of ρ_i, $i = 1, 2, 3, 4$, for a given robot facing the following

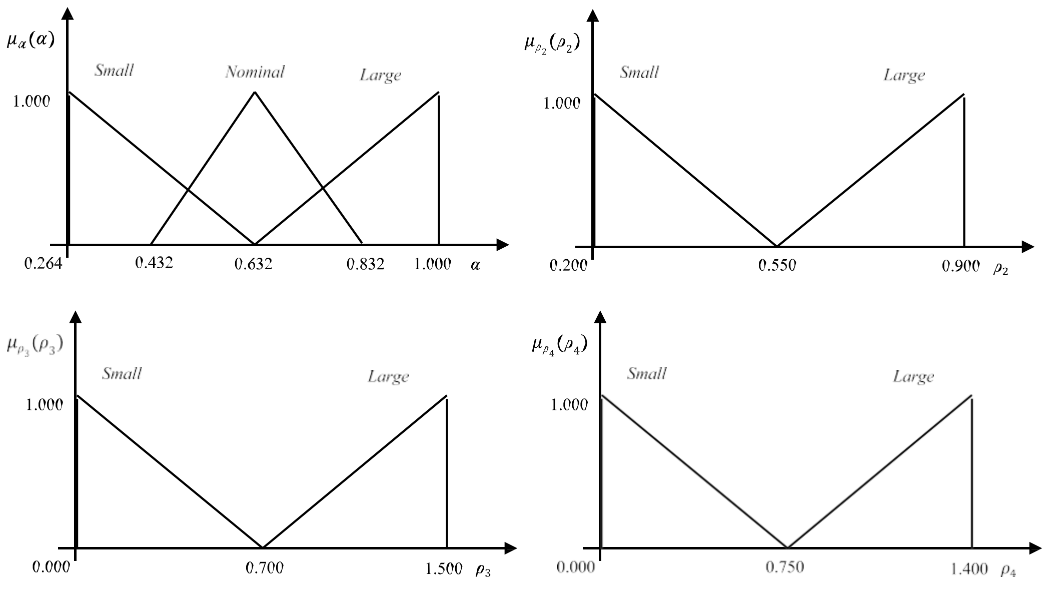


Fig. 11. Membership functions to quantify the consequents for each coefficient.

IF $A_s[t]$ **IS** *Active* **THEN** α **IS** *Small*
  ELSE α **IS** *Large*
IF $O_n[t]$ **IS** *Close* **OR** $P_n[t]$ **IS** *Far* **OR** $C_s[t]$ **IS NOT** *Connected* **THEN** α **IS** *Nominal*
IF $S_n[t]$ **IS** *Social* **OR** $O_n[t]$ **IS** *Close* **OR** $P_n[t]$ **IS** *Far* **OR** $C_s[t]$ **IS NOT** *Connected* **THEN** ρ_2 **IS** *Small*
  ELSE ρ_2 **IS** *Large*
IF $O_n[t]$ **IS** *Close* **THEN** ρ_3 **IS** *Large*
  ELSE-IF $P_n[t]$ **IS** *Far* **OR** $C_s[t]$ **IS NOT** *Connected* **THEN** ρ_3 **IS** *Small*
IF $P_n[t]$ **IS** *Far* **OR** $C_s[t]$ **IS NOT** *Connected* **THEN** ρ_4 **IS** *Large*
  ELSE-IF $O_n[t]$ **IS** *Close* **THEN** ρ_4 **IS** *Small*

Fig. 12. Set of *IF-THEN-ELSE* fuzzy rules do control robots' behavior based on contextual information.

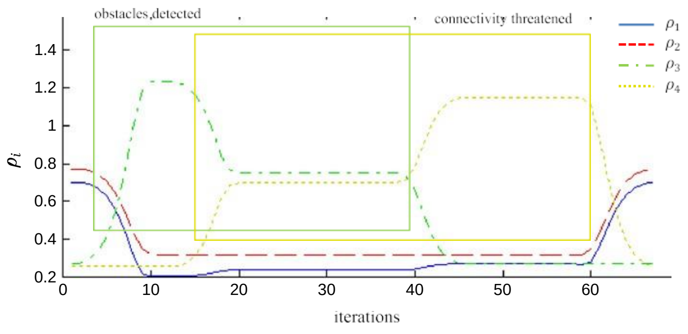


Fig. 13. Parameters' evolution under a hypothetical situation.

situations: (i) the robot is traveling until it first detects an obstacle (i.e., ρ_3 increases); (ii) while still facing the obstacle, the robot moves too far away from its closest neighbor (i.e., ρ_4 increases); (iii) the robot is able to circumvent the obstacle being still far from its closest neighbor (i.e., ρ_3 decreases and ρ_4 increases); and (iv) the robot is finally able to reduce the distance to its closest neighbor (i.e., $\rho_4$decreases).

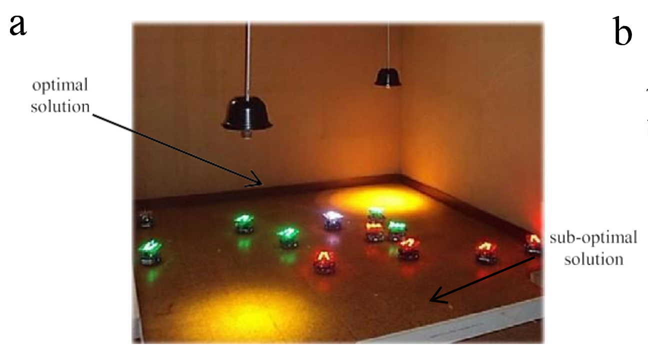
The next section presents experimental results obtained using physical and simulated robots wherein the adaptive version of the *RDPSO* was evaluated and compared to the nonadaptive one.

6. Experimental results

To demonstrate the utility of the proposed distributed adaptive search algorithm, a set of experimental results with multiple simulated and real robots is presented.

6.1. Hardware experiments

In this sub-section, the effectiveness of using the *RDPSO* on swarms of *eSwarBots* [28] is explored, while performing a collective search task under communication constraints. Since the *RDPSO* is a stochastic algorithm, it may lead to a different trajectory convergence whenever it is executed. Therefore, test groups of 30 trials of 180 s each were considered for $N = 12$ *eSwarBots* and an initial number of 2 swarms. The maximum traveled distance between iterations is set to 0.20 m, i.e., $\max|x_n[t+1] - x_n[t]| = 0.20$ while the maximum communication distance between robots is set to $d_{\max} = 1$ m. Inter-robot communication to share positions and individual solutions is carried out using *ZigBee* 802.15.4 wireless protocol. Since *eSwarBots* are equipped with *XBee* modules, that allow a maximum communication range larger than the whole scenario, robots are provided with a list of their



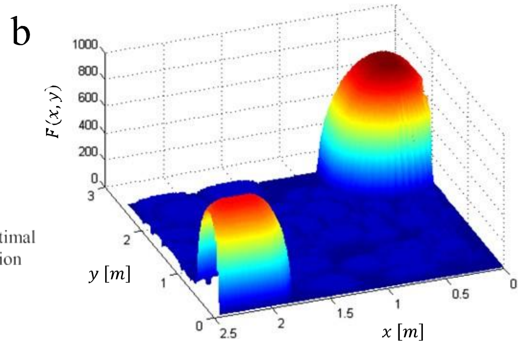


Fig. 14. Experimental setup. (a) Arena with 2 swarms (different colors) of 6 *eSwarBots* each; (b) Virtual representation of the target distribution.

Table 2
Parameters of the nonadaptive *RDPSO*.

S	α	ρ_1	ρ_2	ρ_3	ρ_4
1	0.632	0.100	0.300	0.790	0.790

teammates' address in order to simulate an ad-hoc multi-hop network communication with limited range. At each trial, robots are manually deployed on the scenario in a spiral manner (cf., [9]) while preserving the maximum communication distance $d_{\max}$.

The experimental environment consists of a scenario with dimensions 2.55 m $\times$ 2.45 m, involving obstacles randomly deployed at each trial (Fig. 14(a)). It should be noted that a population of 15 robots corresponds to a density of more than 2 $\text{robot}.\text{m}^{-2}$. As Fig. 14(b) depicts, the objective function is represented by a sub-optimal and an optimal solution (brighter site). The intensity values $F(x,y)$ represented in Fig. 14(b) were obtained sweeping the whole scenario with a single robot in which the light sensor was connected to a 10-bit analog input, thus offering a resolution of approximately 5 mV. To improve the interpretation of the algorithm performance, results were normalized in a way that the objective of robotic teams is to find the optimal solution of $f(x,y) = 1$.

The adaptive *RDPSO* was compared with the nonadaptive *RDPSO* in which parameters presented in Table 2 were chosen in order to satisfy the conditions previously presented in (3) and (4).

Since these experiments represent a search task, it is necessary to evaluate both the completeness of the mission and the time needed to conclude it. Fig. 15 depicts the convergence of both nonadaptive and adaptive *RDPSO* for the proposed conditions. The colored zones between the darker lines represent the interquartile range (i.e., midspread) of the best solution in the 30 trials that was taken as the final output for each different condition for $N = 12$ robots. In other words, the lower line corresponds to the first quartile (i.e., splits lowest 25% of data), the middle one to the second quartile (i.e., median value) and the upper line to the third quartile (i.e., splits highest 25% or lowest 75% of data).

As one may observe, the adaptive version of the *RDPSO* improves the convergence rate of the algorithm also marginally improving the median value of the solution at the end of the mission, i.e., time $=$ 180 s. On the other hand, the nonadaptive *RDPSO* presents a more inconsistent solution, i.e., the interquartile range represented by the stripped red area is larger than the one represented by the solid blue area. However, analyzing swarm algorithms within small populations of 12 robots may not represent the required collective performance (cf., [4]). Also, it may not be enough to assess the *RDPSO* performance within the small proposed scenario. Hence, next section presents computational experiments using a larger population of simulated robots within a larger scenario.

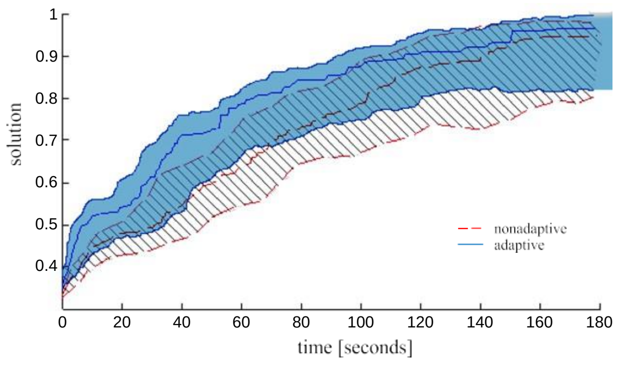


Fig. 15. Performance of the nonadaptive and adaptive *RDPSO* for a population of 12 physical robots.

6.2. *Virtual environment*

The use of simulated robots instead of the physical ones was necessary to further evaluate the adaptive *RDPSO* within large populations of robots within larger scenarios. All of the experiments were carried out in a simulated scenario of 600×600 m with obstacles randomly deployed at each trial. The benchmark function $F(x,y)$ was defined as a common *Gaussian* function (Fig. 17(a)) normalized as:

$$f(x,y) = \frac{F(x,y) - \max F(x,y)}{\min F(x,y) - \max F(x,y)} \quad (14)$$

where x and y-axis represent the planar coordinates in meters. Hence, the objective of robotic teams is to maximize $f(x,y)$, that is, to minimize the original benchmark functions $F(x,y)$, thus finding the optimal solution of $f(x,y) = 1$, while avoiding obstacles and ensuring the *MANET* connectivity.

Test groups of 100 trials and 500 iterations each were considered for $N = \{25, 50, 100\}$ robots. Also, a minimum, initial and maximum number of 2, 5 and 8 swarms were used. The maximum traveled distance between iterations was set as 0.750 m, i.e., $\max|x_n[t+1] - x_n[t]| = 0.750$ while the maximum communication distance between robots was set to $d_{\max} = 15$ m.

Fig. 16 depicts the convergence of both nonadaptive and adaptive *RDPSO* in which the median, first and third quartiles of the best solution in the 100 experiments was taken as the final output in the set $N = \{25, 50, 100\}$ robots.

Analyzing Fig. 16, it is clear that the proposed mission can be accomplished by any number of robots greater or equal to 25. In fact, independently on the number of robots, both nonadaptive and adaptive *RDPSO* converge to the solution most of the time. Nevertheless, the nonadaptive algorithm presents a larger area (stripped red area) between the first quartile and the median value, especially for a larger number of robots. This means that the nonadaptive algorithm sometimes fails in finding the optimal

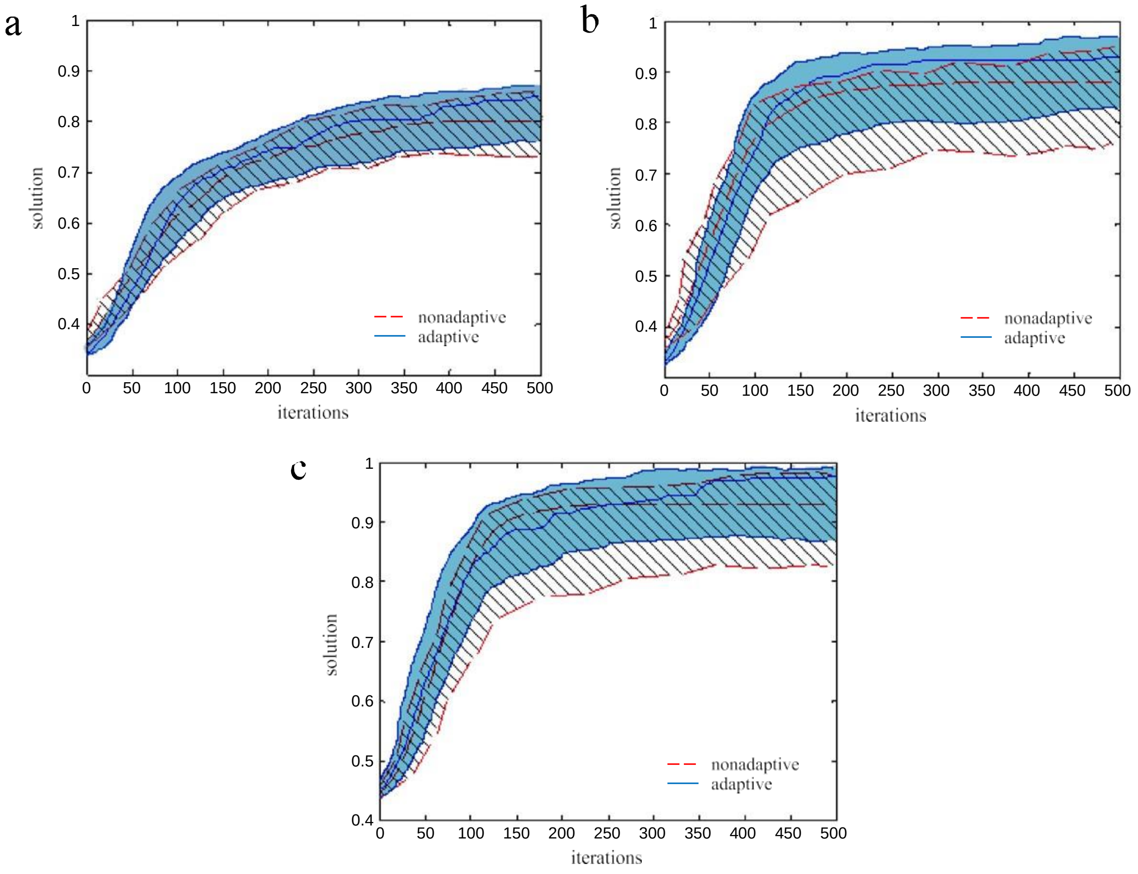


Fig. 16. Performance of the nonadaptive and adaptive *RDPSO* for: (a) $N = 25$ robots; (b) $N = 50$ robots; (c) $N = 100$ robots.

solution getting near the vicinities of it. Also, one can easily observe that the success of the algorithm increases as the number of robots increase, i.e., median value near 1. It can also be observed that the adaptive strategy slightly improves the convergence of the *RDPSO*. Nevertheless, it is still not clear if such improvement is substantial. Hence, to further improve the comparison between nonadaptive and adaptive strategies, heat maps were used (Fig. 17). Heat maps can be designed to indicate how robots tend to be grouped together as well as reflecting the overall quality of the teams.

Fig. 17 presents the heat map of evolutionary trajectories over the 100 trials of 500 iterations each for both nonadaptive and adaptive *RDPSO* for each population of $N = 25, 50, 100$. The hot (i.e., darkest) colors denote the regions in which robots tend to focus their attention, i.e., the most visited regions. Notwithstanding on the nonadaptive or adaptive *RDPSO* nor the number of robots within the population, the most visited regions correspond to the sub-optimal and optimal solutions. Despite the *RDPSO* being able to find the optimal solution in most of the situations (cf., Fig. 15), it is still possible to observe that robots also converge to sub-optimal solutions before being able to ultimately converge to the optimal one (due to social exclusion behavior). It may also be observed that the adaptive *RDPSO* presents a larger diversity (i.e., higher exploration behavior), as the colored regions are larger than those of the nonadaptive case. This is a foreseeable influence of the systematic adjustment of the *RDPSO* parameters. On the other hand, the adaptive *RDPSO* is also able to combine this exploration behavior with a high level of exploitation as the hot colors are more concentrated into smaller circles when compared to the nonadaptive case. In summary, one can observe that both nonadaptive and adaptive algorithms present a high efficiency since the intrinsic features of the *RDPSO* (i.e., social exclusion and inclusion) allows avoiding sub-optimal solutions in most situations. Therefore, to further compare both approaches, a dynamically changing environment is considered.

Due to the continual changes of such environments, the optimal solution in the environment will also change over time. This demands that the *RDPSO* needs to be able not only find the solution in a short time, but also track the trajectory of the optimal solution in the dynamic environment. Nonadaptive algorithms, such as the regular *RDPSO*, usually present several drawbacks in dynamic problems since they lack the ability to track the nonstationary optimal solution in the dynamically changing environment (e.g., [37,38]).

Chaotic functions are the most common and well-studied way to generate nonstationary functions (e.g., logistic functions [39]). In this work, a general way to dynamically change the peaks location based on *Forced Duffing Oscillator* is used [40]. Hence, the function $F(x, y)$ is defined as a dynamic Gaussian function that changes over time based on the algorithm presented in Appendix B. A sequence of the $F(x, y)$ peaks' motion is represented in Fig. 18. The motion of each peak can be configured through the tuple $\{\gamma, \omega, \varepsilon, \Gamma, \Omega\}$ where γ controls the size of the damping, ω controls the size of the restoring force, ε controls the amount of nonlinearity in the restoring force, Γ controls the amplitude of the periodic driving force and Ω controls the frequency of the periodic driving force (cf., Appendix B). Although the tuple $\{\gamma, \omega, \varepsilon, \Gamma, \Omega\}$ may be randomly defined for a more unexpected and chaotic behavior, to better understand the experimental results, it was defined with the constants $\{0.1, 1, 0.25, 1, 0.5\}$. To soften the surface, a circular averaging filter is also applied.

Similarly as before, Fig. 19 depicts the performance of the nonadaptive and adaptive *RDPSO* under a dynamic environment. Once again, analyzing Fig. 19, it is clear that the proposed mission can be accomplished by any number of robots greater or equal

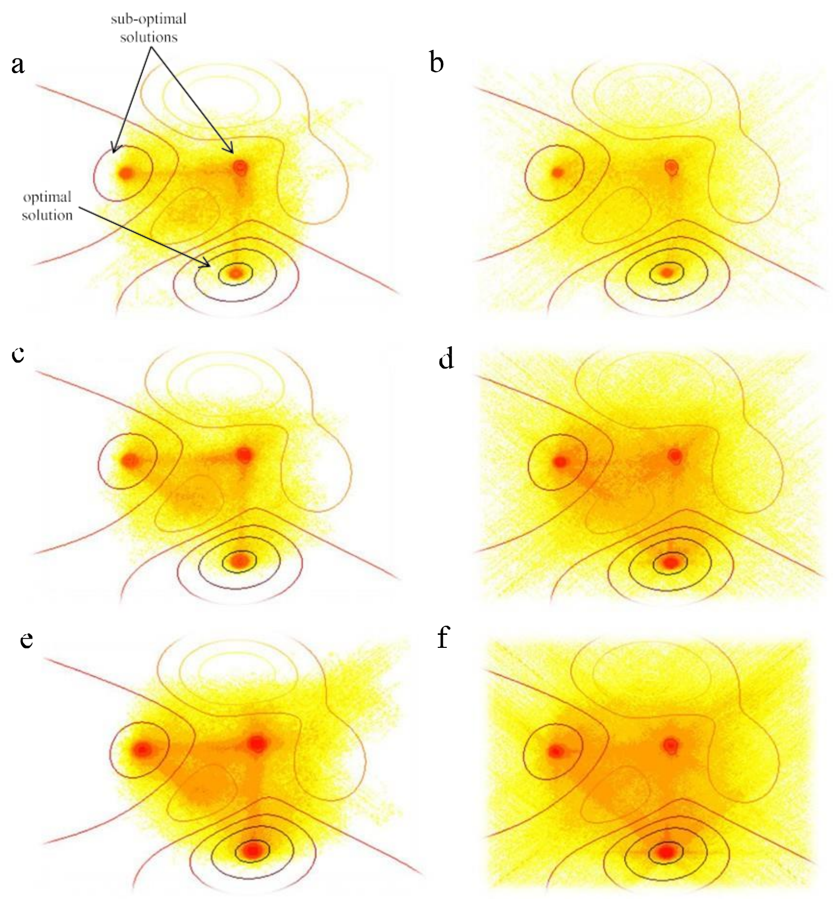


Fig. 17. Heat maps representation of robots' trajectories. (a) Nonadaptive $N = 25$; (b) Adaptive $N = 25$; (c) Nonadaptive $N = 50$; (d) Adaptive $N = 50$; (e) Nonadaptive $N = 100$; (f) Adaptive $N = 100$.

than 25. However, one can observe that the adaptive strategy improves the convergence of the *RDPSO*. That difference is more visible than in the previous static example, both for the median value and the variability of the solution. In the adaptive *RDPSO*, this last one is lower than the nonadaptive *RDPSO* and the difference increases as the number of robots increases. It is also clear that the nonadaptive *RDPSO* seems to be unable to always successfully track the optimal solution, thus increasing the inconsistency of the final result obtained (larger interquartile range). Moreover, it is interesting to observe that, in the adaptive *RDPSO*, the line representing the third quartile (top solid blue line) gets closer to the one representing the median value (darker solid blue line). In other words, the data distribution turns out to be negatively skewed (i.e., the mean is smaller than the median). This means that, in this case, as the goal is to maximize the normalized objective function, approximately 50% of the trials are around the desired objective value for the adaptive *RDPSO* under a dynamic environment.

To further improve the comparison between nonadaptive and adaptive strategies, heat maps were used (Fig. 20). The blue arrows represent the trajectory carried out by the sup-optimal and optimal solutions during the 500 iterations. Fig. 20 presents the heat map of evolutionary trajectories over the 100 trials of 500 iterations each for both nonadaptive and adaptive *RDPSO*, under a dynamic objective function, for each population of $N = 25, 50, 100$. Although the most visited regions correspond to the vicinities of the solutions, the algorithm is unable to effectively track the exact trajectory in some specific situations (cf., Fig. 20(a)). Once again, one may observe that the adaptive *RDPSO* present a higher exploration behavior keeping a high level of exploitation as the hot (i.e., darkest) colors are more concentrated around the solutions' trajectories.

7. Conclusion

The previously proposed Robotic Darwinian Particle Swarm Optimization (*RDPSO*) algorithm is a sociobiologically inspired parameterized swarm algorithm that takes into account real-world multi-robot system characteristics. This paper presented an extension of the *RDPSO* with adaptive capabilities based on contextual information. To that end, a swarm of two physical platforms was used to evaluate constraints such as robot dynamics, obstacles and communication, thus allowing defining metricsat the micro and macro level. Afterward, those context-basedmetrics were used as inputs of a fuzzy system to systematically adapt the *RDPSO* algorithms. Experimental results show that the adaptive version of the algorithm presents an improved convergence when compared to the traditional one on both real and simulated trials. Also, the distribution of target locations, i.e., main objective function, does not greatly affect the adaptive algorithm performance. Even within a dynamic distribution, the adaptive *RDPSO* is able to track the optimal solution easier than the nonadaptive case. As future work, this novel adaptive *RDPSO* will be compared to other swarm robotic algorithms within larger population of real platforms.

Acknowledgments

This work was supported by a *Ph.D.* scholarship (*SFRH*/*BD*/ 73382/2010) granted to the first author by the Portuguese

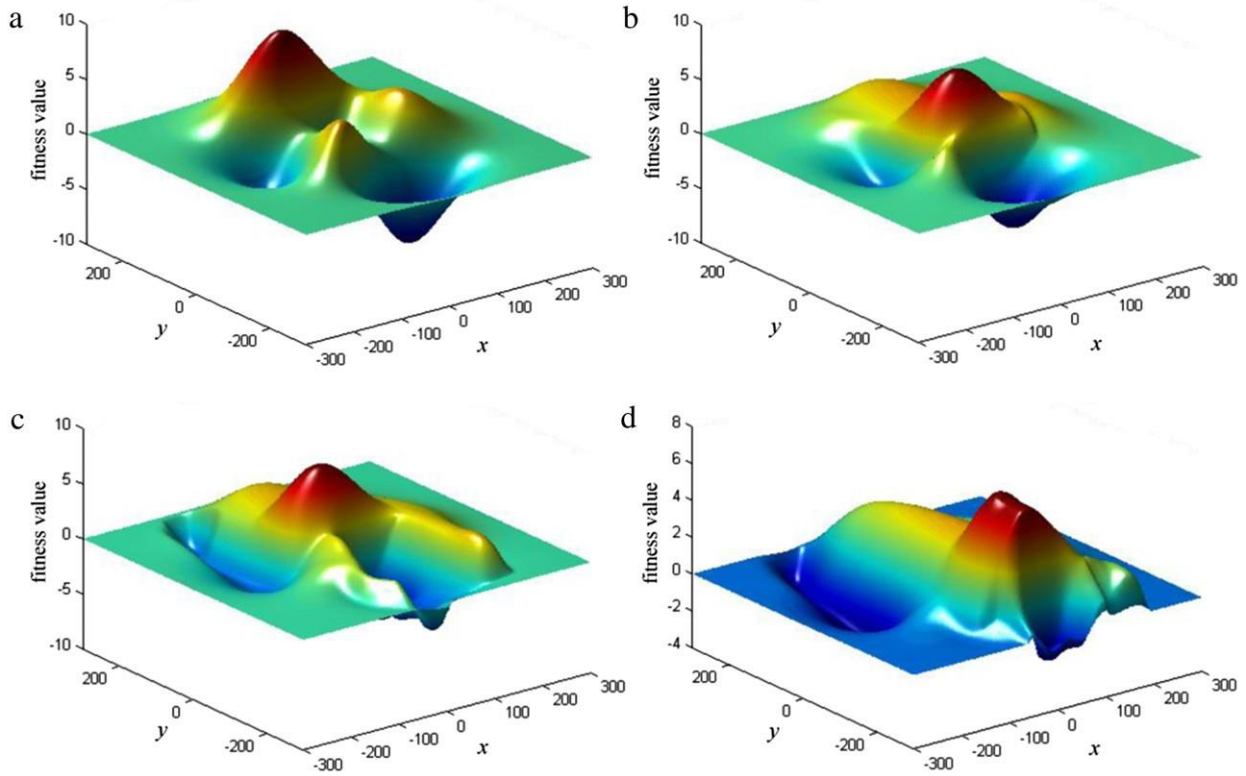


Fig. 18. Planar motion of $F(x, y)$ peaks based on Forced Duffing Oscillator. (a) $t = 0$; (b) $t = 150$; (c) $t = 300$; (d) $t = 450$ iterations.

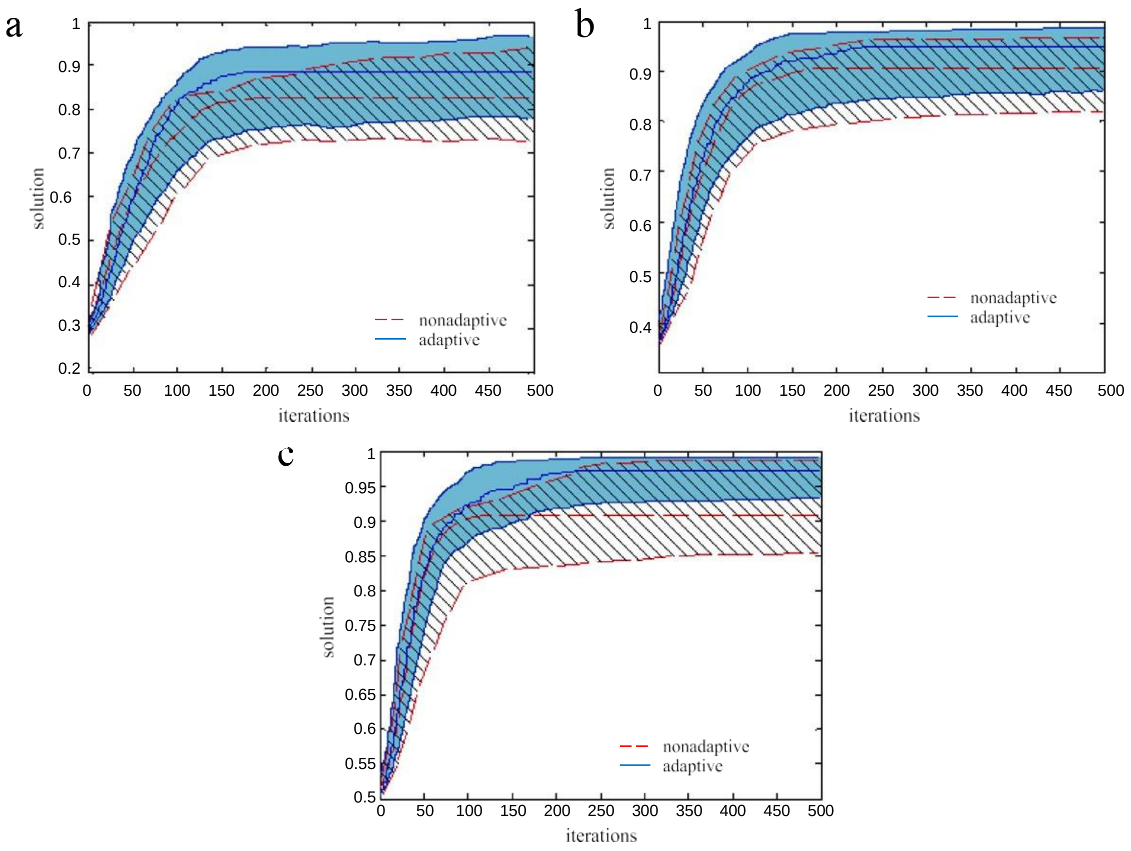


Fig. 19. Performance of the nonadaptive and adaptive *RDPSO* under a dynamic environment for: (a) $N = 25$ robots; (b) $N = 50$ robots; (c) $N = 100$ robots.

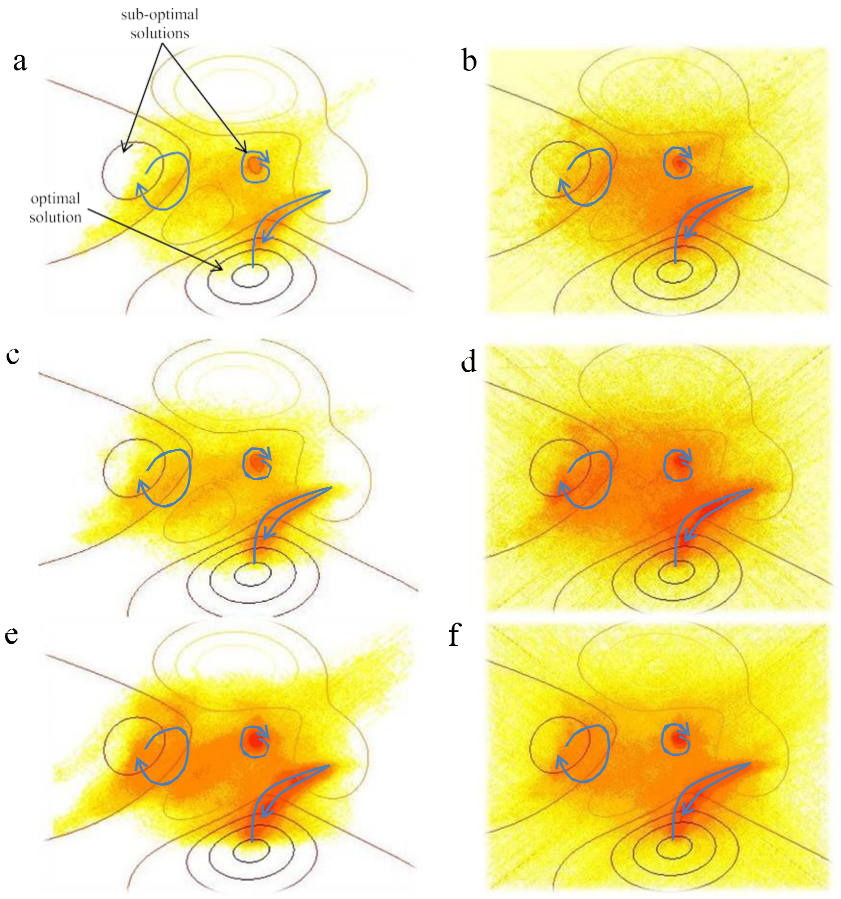


Fig. 20. Heat maps representation of robot's trajectory in a dynamic environment. The blue arrows indicate the trajectory of the sub-optimal and optimal solutions. (a) Nonadaptive $N = 25$; (b) Adaptive $N = 25$; (c) Nonadaptive $N = 50$; (d) Adaptive $N = 50$; (e) Nonadaptive $N = 100$; (f) Adaptive $N = 100$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Foundation for Science and Technology (*FCT*), and by the Institute of Systems and Robotics (*ISR*) and the research project CHOPIN (PTDC/EEA-CRO/119000/2010) also under regular funding by FCT. This work was made possible by the support and assistance of Carlos Figueiredo, Miguel Luz and Monica Ivanova for their cooperation and advice that were vital to fulfill the experiments and other works of technical nature. Furthermore, the authors would also like to thank the referees for all their helpful feedback.

Appendix A

The real polynomial $p(\lambda)$ described in Eqs. (1) and (2) can be rewritten as:

$$a_0\lambda^5 + a_1\lambda^4 + a_2\lambda^3 + a_3\lambda^2 + a_4\lambda + a_5 = 0. \tag{15}$$

Furthermore, one can construct an array having initial rows defined as:

$$\begin{aligned} &\{c_{11}, c_{12}, \ldots, c_{1,6}\} = \{a_0, a_1, \ldots, a_5\}, \\ &\{d_{11}, d_{12}, \ldots, d_{1,6}\} = \{a_5, a_4, \ldots, a_0\}, \end{aligned} \tag{16}$$

and subsequent rows defined by:

$$c_{\beta\gamma} = \begin{vmatrix} c_{\beta-1,1} & c_{\beta-1,\gamma+1} \\ d_{\beta-1,1} & d_{\beta-1,\gamma+1} \end{vmatrix}, \tag{17}$$

$$d_{\beta\gamma} = c_{\beta,8-\gamma-\beta}, \tag{18}$$

where $\beta = 2, 3, 4, 5, 6$ and $\gamma = 0, 1, 2, 3$.

Jury–Marden's Theorem [27] considers that all roots of polynomial $p(\lambda)$ have modulus less than one if and only if $d_{21} > 0$, $d_{T1} < 0$, for $T = 3, 4, 5, 6$. Hence, solving this conditions results in (3) and (4).

Appendix B

Algorithm I. Dynamic function generator

```
$[X,Y,Z] = generate\_surface()$ // the initial surface depends on user's choice; for instance, the MatLab **peaks()** function returns a Gaussian distribution with 5 peaks (3 with positive amplitude and 2 with negative amplitude)
$[z_i, z_j] = find(|Z| > z_T)$ // detect the peaks; the threshold depends on the peaks' maximum amplitude – 20% of the absolute maximum amplitude may be an option
$Z_b = zeros(size(Z))$ // initialize the binary surface as a matrix of zeros
$Z_b(z_i, z_j) = 1$ // the surface peaks are identified by logical ones
$[B,L,N] = boundaries(Z_b)$ // detect the region boundaries in the binary surface; for instance, the MatLab **bwboundaries()** function returns the label matrix $L$ as the second output argument and the number of peaks $N$ as the third argument.
For $i = 1{:}N$
    $[x_P\{i\}, y_P\{i\}] = find(L == i)$ // detect the peaks; the threshold depends on the peaks' maximum amplitude – 20% of the absolute maximum amplitude may be an option
    $\dot{x}\{i\}(t) = -\gamma x(t) + \omega^2 x(t) - \varepsilon x(t)^3 + \Gamma\cos(\Omega t)$ // chaotic motion based on Forced Duffing Oscillator; one can use the MatLab **ode45()** to solve this differential equation problem
    $\dot{y}\{i\}(t) = x(t)$ // the $\{\gamma, \omega, \varepsilon, \Gamma, \Omega\}$ tupple may be randomly defined for each peak
$Z_P = Z$ // initialize the modified surface as the original one
For $t = 1{:}T$ // the total time (i.e., number of iterations) of the motion is defined by $T$
    For $i = 1{:}N$
        $Z_{aux} = Z_P(x_P\{i\}, y_P\{i\})$ // save the current peak in a temporary variable
        $x_P\{i\} = x_P\{i\} + round(x\{i\}(t))$ // the elements of the matrix needs to be integer values
        $y_P\{i\} = y_P\{i\} + round(y\{i\}(t))$
        $Z_P(x_P\{i\}, y_P\{i\}) = Z_{aux}$ // new position of the peak
    $Z_P = filter(Z_P)$ // a filter may be used for a more soften surface; the MatLab **filter2()** and **fspecial()** functions may be used for that purpose
    $visualize(X,Y,Z_P)$ // optionally, one can visualize the new surface; the MatLab **surf()** function may be used
```

References

[1] K.M. Passino, Biomimicry for Optimization, Control, and Automation, Springer-Verlag, London, 2005.

[2] M.S. Couceiro, N.M.F. Ferreira, J.A.T. Machado, Hybrid adaptive control of a dragonfly model, Communications in Nonlinear Science and Numerical Simulation 17 (2) (2012) 893–903.

[3] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, Oxford University Press, New York, 1999.

[4] G. Beni, From swarm intelligence to swarm robotics, in: Proceedings of the Swarm Robotics Workshop, Heidelberg, Germany, 2004, pp. 1–9.

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

[6] J. Tillett, T.M. Rao, F. Sahin, R. Rao, S. Brockport, Darwinian particle swarm optimization, in: Proceedings of the 2nd Indian International Conference on Artificial Intelligence, 2005, pp. 1474–1487.

[7] M.S. Couceiro, N.M.F. Ferreira, J.A.T. Machado, Fractional order Darwinian particle swarm optimization, in: 3th Symposium on Fractional Signals and Systems, FSS'2011, Coimbra, Portugal, 2011.

[8] M.S. Couceiro, R.P. Rocha, N.M.F. Ferreira, A novel multi-robot exploration approach based on particle swarm optimization algorithms, in: IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR2011, Kyoto, Japan, 2011.

[9] M.S. Couceiro, R.P. Rocha, N.M.F. Ferreira, Ensuring Ad Hoc connectivity in distributed search with Robotic Darwinian swarms, in: Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR2011, Kyoto, Japan, 2011, pp. 284–289.

[10] J. Suarez, R. Murphy, A survey of animal foraging for directed, persistent search by rescue robotics, in: Proceedings of the 2011 IEEE International Symposium on Safety, Security and Rescue Robotics, Kyoto, Japan, 2011, pp. 314–320.

[11] L. Marques, U. Nunes, A.T. de Almeida, Particle swarm-based olfactory guided search, Autonomous Robots 20 (3) (2006) 277–287.

[12] M. Clerc, J. Kennedy, The particle swarm—explosion, stability, and convergence in a multidimensional complex space, IEEE Transactions on Evolutionary Computation 6 (1) (2002) 58–73.

[13] V. Kadirkamanathan, H. Selvarajah, P.J. Fleming, Stability analysis of the particle dynamics in particle swarm optimizer, IEEE Transactions on Evolutionary Computation 10 (3) (2006) 245–255.

[14] K. Yasuda, N. Iwasaki, G. Ueno, E. Aiyoshi, Particle swarm optimization: a numerical stability analysis and parameter adjustment based on swarm activity, IEEJ Transactions on Electrical and Electronic Engineering 3 (2008) 642–659. Wiley InterScience.

[15] L.A. Zadeh, Fuzzy Sets, Information and Control 8 (1965) 338–353.

[16] Y. Shi, R. Eberhart, Fuzzy adaptive particle swarm optimization, in: Proc. IEEE Congr. Evol. Comput., vol. 1, 2001, pp. 101–106.

[17] H. Liu, A. Abraham, W. Zhang, A fuzzy adaptive turbulent particle swarm optimisation, International Journal of Innovative Computing and Applications 1 (1) (2007) 39–47.

[18] T. Blackwell, P. Bentley, Don't push me! collision-avoiding swarms, in: Proc. IEEE Congr. Evol. Comput., vol. 2, 2002, pp. 1691–1696.

[19] T. Krink, J. Vesterstrom, J. Riget, Particle swarm optimization with spatial particle extension, in: Proc. IEEE Congr. Evol. Comput., vol. 2, 2002, pp. 1474–1479.

[20] V. Miranda, N. Fonseca, New evolutionary particle swarm algorithm, EPSO, applied to voltage/VAR control, in: Proc. 14th Power Syst. Comput. Conf., 2002.

[21] M. Lovbjerg, T. Krink, Extending particle swarms with self-organized criticality, in: Proc. IEEE Congr. Evol. Comput., vol. 2, 2002, pp. 1588–1593.

[22] D. Saikishan, K. Prasanna, Multiple robot co-ordination using particle swarm optimisation and bacteria foraging algorithm, B.Tech. Thesis, Department of Mechanical Engineering, National Institute of Technology, 2010.

[23] J. Hereford, M. Siebold, Multi-robot search using a physically-embedded particle swarm optimization, International Journal of Computational Intelligence Research 4 (2) (2008) 197–209.

[24] R.M. Turner, Context-mediated behavior for intelligent agents, International Journal of Human–Computer Studies 48 (1998) 307–330.

[25] D. Calisi, L. Iocchi, D. Nardi, G. Randelli, V.A. Ziparo, Improving search and rescue using contextual information, Advanced Robotics 23 (2009) 1199–1216.

[26] M.S. Couceiro, F.M.L. Martins, R.P. Rocha, N.M.F. Ferreira, Analysis and parameter adjustment of the RDPSO—towards an understanding of robotic network dynamic partitioning based on Darwin's theory, International Mathematical Forum 7 (32) (2012) 1587–1601. Hikari, Ltd.

[27] S. Barnett, Polynomials and Linear Control Systems, Marcel Dekker, Inc., New York, USA, 1983.

[28] M.S. Couceiro, C.M. Figueiredo, J.M.A. Luz, N.M.F. Ferreira, R.P. Rocha, A low-cost educational platform for swarm robotics, International Journal of Robots, Education and Art (2011).

[29] M.S. Couceiro, R.P. Rocha, C.M. Figueiredo, J.M.A. Luz, N.M.F. Ferreira, Multi-robot foraging based on Darwin's survival of the fittest, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'2012, Vilamoura, Algarve, 2012.

[30] Y. Wakasa, K. Tanaka, Y. Nishimura, Control-theoretic analysis of exploitation and exploration of the PSO algorithm, in: IEEE International Symposium on Computer-Aided Control System Design, IEEE Multi-Conference on Systems and Control, Yokohama, Japan, 2010.

[31] G. Venter, Particle swarm optimization, in: Proceedings of 43rd AIAA/ASME/ASCE/AHS/ASC Structure, Structures Dynamics and Materials Conference, 2002, pp. 22–25.

[32] K. Suresh, et al. Inertia-adaptive particle swarm optimizer for improved global search, in: Eighth International Conference on Intelligent Systems Design and Applications, ISDA'08, Kaohsiung, 2008, pp. 253–258.

[33] N. Michael, M.M. Zavlanos, V. Kumar, G.J. Pappas, Maintaining connectivity in mobile robot networks, in: The 11th International Symposium Experimental Robotics, STAR 54, 2009, pp. 117–126.

[34] D. Portugal, R.P. Rocha, Decision methods for distributed multi-robot patrol, in: Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR'2012, Texas, 2012.

[35] J.M. Garibaldi, E.C. Ifeachor, Application of simulated Annealing Fuzzy Model Tuning to Umbilical Cord Acid–base Interpretation, IEEE Transactions on Fuzzy Systems 1 (7) (1999).

[36] D. Ruan, E.E. Kerre, On if-then-else inference rules, in: IEEE International Conference on Systems, Man, and Cybernetics, 1996, Beijing, China, 2002, pp. 1420–1425.

[37] A. Carlisle, G. Dozier, Adapting particle swarm optimization to dynamic environments, in: Proceedings of the International Conference on Artificial Intelligence, Las Vegas, USA, 2000, pp. 429–433.

[38] X. Cui, T.E. Potok, Distributed adaptive particle swarm optimizer in dynamic environment, in: IEEE International Parallel and Distributed Processing Symposium, IPDPS'07, Long Beach, CA, 2007, pp. 1–7.

[39] R.W. Morrison, K.A. De Jong, A test problem generator for non-stationary environments, in: Proceedings of the 1999 Congress on Evolutionary Computation, CEC'99, Washington DC, USA, 1999.

[40] C.A. Tan, B.S. Kang, Chaotic motions of a Duffing oscillator subjected to combined parametric and quasiperiodic excitation, International Journal of Nonlinear Sciences and Numerical Simulation 2 (4) (2001) 353–364.