Multi-Robot Foraging based on Darwin’s Survival of the Fittest

Micael S. Couceiro, IEEE Student Member, Rui P. Rocha, Carlos M. Figueiredo, J. Miguel A. Luz, N. M. Fonseca Ferreira

Abstract — This paper presents a collective foraging algorithm designed to simulate natural selection in a group of swarm robots. The Robotic Darwinian Particle Swarm Optimization (RDPSO) previously proposed is improved using fractional calculus theory and evaluated on real low-cost mobile robots performing a distributed foraging task. This work aims at evaluating this novel exploration strategy, by studying the performance of the algorithm within a population of up to 12 robots, under communication constraints. In order to simulate the maximum allowed communication distance, robots were provided with a list of their teammates’ addresses. Experimental results show that only 4 robots are needed to accomplish the proposed mission and, independently on the number of robots, maximum communication distance and fractional coefficient, the optimal solution is achieved in approximately 90% of the experiments.

I. INTRODUCTION

The cooperative foraging observed in nature has been studied for the last decades and used as inspiration in optimization problems, applied computation, robotics and many other applications [1]. This collective behavior emerges not only in hunting practices of lions, hyenas or wolves but also in small colonies of insects such as ants, bees, termites or wasps [2]. In spite of the potential advantages of Multi-Robot Systems (MRS) in cooperative foraging tasks, related with space and time distribution, it is necessary that each robot maintains a sufficient and consistent level of awareness about the mission assigned to the team and about its teammates in order to attain effective cooperation while avoiding collisions. Furthermore, in most real situations, robots have to move to complete their tasks while maintaining communication among them using ad-hoc communication, since communication infrastructure may be damaged or missing (e.g., hostile environments, search & rescue, disaster recovery). Just like in MRS wherein groups of robots interact to accomplish their goals [3], particle optimization algorithms, such as the well-known Particle Swarm Optimization (PSO) [4] and the Darwinian Particle Swarm Optimization (DPSO) [5], use groups of interacting virtual agents in order to achieve their optimization. In a previous work [6], a multi-robot exploration algorithm was proposed and denoted as Robotic Darwinian Particle Swarm Optimization (RDPSO). This evolutionary algorithm was evaluated in simulation environment using Darwin’s natural selection to enhance the ability to escape from sub-optimal solution. Also, an extension of the algorithm was proposed in [7] to guarantee Mobile Ad-Hoc Network (MANET) connectivity and to establish the initial deployment of robots based on the Spiral of Theodorus.

A. Statement of Contributions and Paper Organization

Bearing this idea in mind, a brief review of PSO techniques applied to MRS is presented in section II. This is followed by two key contributions demonstrated through an extension of the Robotic Darwinian PSO (RDPSO) previously presented in [6] and [7]. First, the concept of fractional calculus (FC) is used to control the convergence rate of the RDPSO, while considering the robot dynamical characteristics (section III). Secondly, an hierarchical control architecture of the RDPSO is presented where the Low-Level Control (LLC) handles all the kinematics and dynamics inherent to the robots (section IV). To assess these new features of the RDPSO, this novel distributed algorithm is evaluated in a cooperative foraging task performed by real mobile platforms (section V). The main conclusions are outlined in Section VI.

II. RELATED WORK

In nature, Darwin’s natural selection selects foraging behavior that maximizes lifetime calorie content or minimizes the probability of starvation. In robotics, engineering design favors decision-making algorithms that maximize accumulated value or minimize the probability of not reaching performance targets [8]. A behavior-based strategy to maintain MRS connectivity has been studied in [9]. The authors presented the extension of the Null-Space-based Behavioral approach to control a group of marine vehicles to execute missions such as formation control or cooperative target visiting with communication constraints. This is a promising approach since it would be possible to merge the behaviors of the RDPSO with different priorities in order to define the motion directives of robots. However, design choices concerning how to organize the behaviors by priority represent a higher complexity since these choices derive from practical considerations related to both the mission objective and the physical characteristics of the robotic system.
Similarly to the presented work, a multi-search algorithm inspired by the PSO algorithm is proposed in [10]. However, the algorithm was only evaluated based on Webots simulator and the search scenario was very limited, since only one target was used thus avoiding the effect of sub-optimal solution in the PSO algorithm. The work of Prasanna and Saikishan [11] involves path-planning and coordination between multiple mobile agents in a static-obstacle environment based on the PSO and Bacteria Foraging Algorithm (BFA). As this work uses Darwinian principles in the PSO to avoid getting trapped in sub-optimal solution, 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. Also, one only target and two robots were used thus limiting the evaluation of the proposed algorithm. Hereford and Siebold [12] 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 sub-optimal solutions. Also, collision avoidance and fulfillment of MANET connectivity was not considered.

Our previous works [6] [7] presented successive improvements of the proposed RDPSO to evaluate the convergence of the algorithm under communication constraints while increasing the robot population. Based on this previously proposed algorithm, next section extends the convergence rate of the RDPSO using fractional calculus.

III. FRACTIONAL ORDER RDPSO

In this section, a new method to control the convergence rate of the RDPSO algorithm [6] based on Pires et al. approach to the traditional PSO [13] is introduced. A previous work presented a similar approach conducted to the traditional DSO in [14].

To model the swarm, each particle denoted by n moves in a multidimensional space. A particle is represented by the desired position vector \( x_{t+1} \) and velocity vector \( v_{t+1} \) which are highly dependent on the current position vector \( x_t \), velocity vector \( v_t \), local best vector \( c_t \), global best vector \( s_t \), obstacle susceptibility vector \( T_t \), and MANET connectivity vector \( m_t \). The vectors size depends on the dimension of the multidimensional space.

\[
v^n_t = w v^n_t + \rho_1 r_1 (c^n_t - x^n_t) + \rho_2 r_2 (s^n_t - x^n_t) + \rho_3 r_3 (G^n_t - x^n_t) + \rho_4 r_4 (V^n_t - x^n_t),
\]

\[
x^n_{t+1} = x^n_t + v^n_{t+1},
\]

where the coefficients \( w, \rho_1, \rho_2, \rho_3 \) and \( \rho_4 \) 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, respectively.

\( r_1, r_2, r_3 \) and \( r_4 \) are random vectors wherein each component is generally a uniform random number between 0 and 1. Considering equations (1) and (2), it is noteworthy that robots will tend to converge to the optimal solution. However, although all robots within a swarm agree with the best solution, they must also fulfill the other requirements (i.e., avoid obstacles and maintain a certain distance between neighbors).

Fractional calculus (FC) has attracted the attention of several researchers [15], being applied in various scientific fields, such as engineering, computational mathematics, fluid mechanics, among others.

The Grünwald–Letnikov definition based on the concept of fractional differential with \( a \in C \) of the signal \( x(t) \), is given by

\[
D^a[x(t)] = \lim_{h \to 0} \frac{1}{h^a} \sum_{k=0}^{\infty} (-1)^k \Gamma(a+1) x(t-kh) \Gamma(k+1) / (a+k+1),
\]

where \( \Gamma \) is the gamma function. An important property revealed by (3) is that while an integer-order derivative just implies a finite series, the fractional-order derivative requires an infinite number of terms. Therefore, integer derivatives are “local” operators while fractional derivatives have, implicitly, a “memory” of all past events. However, the influence of past events decreases over time.

Based on (3), a discrete time implementation expression can be defined as:

\[
D^a[x(t)] = \frac{1}{T} \sum_{k=0}^{r} \frac{(-1)^k \Gamma(a+1) x(t-kT)}{\Gamma(k+1) / (a+k+1)},
\]

where \( T \) is the sampling period and \( r \) is the truncation order.

The features inherent to fractional calculus make this mathematical tool well suited to describe many phenomena, such as irreversibility and chaos, because of its inherent memory property. In this line of thought, the dynamic phenomena of a robot’s trajectory configure a case where fractional calculus tools fit adequately.

Considering the inertial influence in (1), \( w = 1 \), assuming \( T = 1 \) and based on [13] work, the following expression can be defined:

\[
D^n a_{t+1} = \rho_1 r_1 (c^n_t - x^n_t) + \rho_2 r_2 (s^n_t - x^n_t) + \rho_3 r_3 (G^n_t - x^n_t) + \rho_4 r_4 (V^n_t - x^n_t).
\]

Preliminary experimental tests on the algorithm presented similar results for \( r \geq 4 \). Furthermore, the computational requirements increase linearly with \( r \), i.e., the fractional order RDPSO present a \( O(r) \) memory complexity. Hence, considering only the first four terms, \( r = 4 \), of differential derivative given by (4), equation (5) can be rewritten as:

\[
v^n_{t+1} = a v^n_t + \frac{1}{2} a v^n_{t-1} + \frac{1}{6} a(1-a) v^n_{t-2} + \frac{1}{24} a(1-a)(2-a) v^n_{t-3} + \rho_1 r_1 (c^n_t - x^n_t) + \rho_2 r_2 (s^n_t - x^n_t) + \rho_3 r_3 (G^n_t - x^n_t) + \rho_4 r_4 (V^n_t - x^n_t).
\]
tion behavior). On the other hand, with a large value of $\alpha$ the robot may be unable to prevent collisions and maintain the connectivity with its teammates (i.e., exploration behavior).

One way to analyze the convergence of the algorithm consists on adjusting the fractional coefficient based on physical mobile robots constraints inherent to their dynamical characteristics. Usually, this is not addressed in the literature concerning the classical PSO methods, since virtual agents (i.e., particles) are not constrained by such behaviors. Hence, let us then suppose that a robot is traveling at a constant velocity such that $v_{t-k}^n = v$ for $k \in N_0$ and it is able to find its equilibrium point in such a way that $x_t^n = \hat{e}_t^n = \hat{s}_t^n = \hat{\theta}_t^n = \bar{m}_t^n$. In other words, the robot converges to the best position such that the local, global, obstacle and MANET vectors are coincident. As a result, the robot needs to decelerate until it stops, i.e., $v > v_{t-1}^n \geq \cdots \geq v_{t+k}^n \geq \cdots \geq 0$. Consequently, equation (6) can be rewritten as:

$$0 \leq v \left( \alpha + \frac{1}{2} \alpha + \frac{1}{6} \alpha (1 - \alpha) + \frac{1}{24} \alpha (1 - \alpha) (2 - \alpha) \right) < v,$$

thus resulting in

$$0 < \alpha \leq 0.632. \quad (8)$$

Therefore, one can conclude that $\alpha = 0.632$ is the boundary of the attraction domain: The RDPSO is stable for $0 < \alpha \leq 0.632$ and unstable for $0.632 < \alpha \leq 1$. As a result of the above analysis, the fractional coefficient can be parameterized in such a way that the system’s convergence can be controlled by taking into account obstacle avoidance and MANET connectivity, without resorting to the definition of any arbitrary or problem-specific parameters. Next section presents the control architecture of the RDPSO considering real heterogeneous mobile robots.

IV. CONTROL ARCHITECTURE OF SWARM ROBOTS

This section presents how the RDPSO algorithm can be used on real robotic platforms. The RDPSO performs all the computation by considering the robots as particles, thus ignoring any kinematic or dynamic inherent to the real platform. In other words, the output is a position vector for each robot (cf. equations 2 and 6). Depending on the robot kinematical and dynamical characteristics, this new position may be achieved from different ways. For instance, a holonomic robot (e.g., omnidirectional drive system), which has the kinematic advantage of allowing continuous translation and rotation in any direction, can move to a desired position regardless of its orientation. However, a non-holonomic robot (e.g., differential drive system) needs to first change its orientation to be aligned with the target and then move forward to the desired position.

In order to achieve a higher level of hardware abstraction, a Low-Level Control (LLC) was introduced in the control architecture (Fig. 1). In this work, and since all robots are non-holonomic, the LLC was designed for the specific structure of the differential-drive robot to make it turn and then follow the desired position vector received by the RDPSO.

As shown in Fig. 1, the output of the RDPSO is given as a reference value to the LLC that considers the kinematical and dynamical characteristics of the robot thus defining the commands to the actuators, e.g., number of pulses for DC motors equipped with encoders.

This two-level control loop organization allows using the RDPSO with different types of robotic systems, neglecting their kinematical and dynamical characteristics. These aspects are considered by the LLC that needs to be properly designed for a specific robotic system.

To our specific situation, i.e., differential-drive robots, the LLC receives the desired position $x_{D_{4t+1}}^n$, which corresponds to a cartesian position $[x_{D_{4t+1}}^n, y_{D_{4t+1}}^n]^T$, and computes the inverse kinematic model based on the following equations:

$$h_{t+1}^n = \sqrt{(v_{1t+1}^n)^2 + (v_{2t+1}^n)^2}, \quad (9)$$
$$\sigma_{t+1}^n = \text{atan} 2(v_{2t+1}^n, v_{1t+1}^n), \quad (10)$$
$$\theta_{t+1}^n = \sigma_{t+1}^n - \sigma_t^n, \quad (11)$$

wherein $v_{1t+1}^n$ and $v_{2t+1}^n$ are obtained using (6). The function $\text{atan} 2$ in (10) is a variant of the trigonometric arctangent function, but accounts for the quadrant in which $\sigma_{t+1}^n$ lies. Note that, $\sigma_t^n$ corresponds to the previous computation of (9) and may be considered zero in the first iteration (i.e., initial orientation of zero degrees). The output of the inverse kinematic model is represented by the rotation $\theta_{t+1}^n$ that the robot needs to perform to be aligned with the desired position and the distance $h_{t+1}^n$ it needs to travel to reach it.

The rotation $\tau_{D_{4t+1}}^n$ and the forward movement $\tau_{D_{2t+1}}^n$ of the differential-drive robot are defined by:

$$\tau_{D_{4t+1}}^n = \tau_{\text{rev}} \cdot \frac{\theta_{t+1}^n}{2\pi} \cdot \frac{\tau_{\text{robot}}}{\tau_{\text{wheel}}}, \quad (12)$$
$$\tau_{D_{2t+1}}^n = \tau_{\text{rev}} \cdot \frac{\theta_{t+1}^n}{2\pi} \cdot \frac{1}{\tau_{\text{wheel}}}, \quad (13)$$

wherein $\tau_{\text{rev}}$ is the total number of steps or pulses per revolution. The radius of the robot and the wheels are defined by $r_{\text{robot}}$ and $r_{\text{wheel}}$, respectively. In order to improve the time response of the robot and the smoothness of its movement, a rotational threshold $\theta_T$ was introduced. Rotations $\theta_{t+1}^n$ inferior to $\theta_T$ are then ignored and only the forward distance $h_{t+1}^n$ is considered. Bearing in mind this assumption, and since a possible loss of steps or pulses may occur while executing the commands, i.e., $\tau_{t+1}^n \neq \tau_{D_{4t+1}}^n$ or $\tau_{t+1}^n \neq \tau_{D_{2t+1}}^n$, a new real position is then recalculated and consid-
ered as the current position $x_{i+1}$ of the robot. This new position and the corresponding value of the objective function $f(x_{i+1})$ defined in this position (i.e., sensed by the sensory system) needs to be shared between robots (cf., Communication System in Fig. 1) so that cooperation can emerge. To that end, this information is sent directly to the robots in the neighborhood defined by $d_{\text{max}}$ and relayed to other robots based on ad-hoc networking paradigm.

V. EXPERIMENTAL RESULTS

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. Multiple test groups of 20 trials of 3 minutes each were considered. A minimum, initial and maximum number of 1, 2 and 3 swarms were used independently of the population of robots.

The eSwarmBot (Educational Swarm Robot) was the platform used to evaluate the algorithm (Fig. 2). It consists on a differential ground platform recently developed and presented in [16] 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 $N = \{4, 8, 12\}$. RGB-LEDs on top of the eSwarmBots are used to identify its swarm. RGB-LEDs allow the representation of a wide range of different colors matching different swarms. All of the experiments were carried out in a 2.55 meters to 2.45 meters scenario.

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, 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 $d_{\text{max}}$ was varied between 0.50 meters and 1.50 meters. At each trial, robots were manually deployed on the scenario in a spiral manner while preserving the maximum communication distance $d_{\text{max}}$ (as previously presented in [7]).

In order to evaluate the impact of fractional calculus in the convergence of the algorithm, the original RDPSO ($\alpha = 1.0$) was compared to the fractional order RDPSO with $\alpha = 0.632$ (cf., Section III).

The previously described conditions give a total of 240 experiments, thus leading to a runtime of 12 hours. The next sequence of frames (Fig. 4) presents a trial of the team’s performance using $N = 12$, $d_{\text{max}} = 1.50$ meters and $\alpha = 0.632$. 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, Fig. 5 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 $d_{\text{max}}$ and $\alpha$.

Analyzing Fig. 5, 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 rate. A population of 4, 8 and 12 robots takes, in average, 77, 106 and 112 seconds to converge to the optimal solution, respectively. This is a consequence 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 optimal solution. On the other hand, the decrease of $a$ from 1 to 0.632 improves the convergence rate of the algorithm in some situations also marginally improving the median value of the solution. However, this relationship is not linear and varies depending on the maximum communication distance and the number of robots in the population. For instance, for a communication distance of 0.5 meters and regardless on the number of robots, the fractional order RDPSO depicts an improved convergence when compared to the integer order. 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, which makes it difficult for the robots to complete the proposed odometry objectives accurately.

VI. CONCLUSION

This paper presents the evaluation of a foraging sociobiological-inspired algorithm, denoted as RDPSO - Robotic Darwinian Particle Swarm Optimization, on real platforms. Furthermore, fractional calculus is introduced to control the convergence rate of the algorithm taking into account robot dynamics. Experimental results show that the performance of the robotic population in the proposed scenario, with a biologically inspired behaviour based on natural selection and social exclusion, is not significantly affected by the number of robots in the population, the maximum communication distance between robots and the fractional coefficient $a$. Yet, the fractional extension of the RDPSO allows taking into account the dynamic phenomena of robots' trajectory. Nevertheless, one of the main limitations of the algorithm resides in having several parameters 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. Also, eSwarBot platforms should be enhanced in order to avoid mislocalization errors. The robots odometry could be improved by replacing the current encoders with more precise and with better resolution ones, keeping the low-cost goal. Nevertheless, this exchange may not be enough by itself. There are other methods that can be used with the current odometry in order to overcome these problems by correcting the errors in robot location based on reference points in the scenario. For instance, analysing the strength of the received signal (RSSI) [17] (taking advantage of this XBee modules' capability) would allow determining the location of mobile module through triangulation methods.
Fig. 5. Performance of the algorithms changing the number of robots $N$ in the population: a) $d_{\text{max}} = 0.5$ meters; $\alpha = 1.0$; b) $d_{\text{max}} = 1.5$ meter; $\alpha = 1.0$; c) $d_{\text{max}} = 0.5$ meters; $\alpha = 0.632$; d) $d_{\text{max}} = 1.5$ meter; $\alpha = 0.632$.

REFERENCES


