Hybrid Training to Generate Robust Behaviour for Swarm Robotics Tasks

Pedro Romano[1,3], Luís Nunes[1,2] [a] and Sancho Oliveira[1,2,3] [b]

[1]*Iscte - Instituto Universitário de Lisboa, Av. Forças Armadas, Lisboa, Portugal*
[2]*ISTAR_Iscte, Lisboa, Portugal*
[3]*Instituto de Telecomunicações, IT_Iscte, Lisboa, Portugal*

Abstract: Training of robotic swarms is usually done for a specific task and environment. The more specific the training is, the more the likelihood of reaching a good performance. Still, flexibility and robustness are essential for autonomy, enabling the robots to adapt to different environments. In this work, we study and compare approaches to robust training of a small simulated swarm on a task of cooperative identification of moving objects. Controllers are obtained via evolutionary methods. The main contribution is the test of the effectiveness of training in multiple environments: simplified versions of terrain, marine and aerial environments, as well as on ideal, noisy and hybrid (mixed environment) scenarios. Results show that controllers can be generated for each of these scenarios, but, contrary to expectations, hybrid evolution and noisy training do not, in general, generate better controllers for the different scenarios. Nevertheless, the hybrid controller reaches a performance level par with specialized controllers in several scenarios, and can be considered a more robust solution.

1 INTRODUCTION

The penetration of fully autonomous robots in society is still scarce. One of the key factors of this challenge is environment perception. In order to behave autonomously, the robot needs to make a wide variety of decisions that have to be supported by a great understanding of the environment surrounding it (Fitzpatrick, 2003).

"Machine Perception" is a term used to describe the capability of a machine to interpret data much like humans use their senses to perceive the world around it. A good level of perception will ultimately boost the level of situation awareness, greatly improving the chances of making a good decision.

Classic methods for synthesizing robotic controllers are based on the manual specification of its behavior. For greater levels of complexity, manually specifying all possible use cases and scenarios a robot may encounter gets specially demanding. This has motivated the application of artificial intelli-

[a] https://orcid.org/0000-0001-7072-0925
[b] https://orcid.org/0000-0003-1391-3194

gence (AI) and evolutionary computation (a subfield of AI and machine learning) to synthesize robotic controllers. This approach started having promising results (Lewis et al., 1992; Cliff et al., 1993) as the evolutionary robotics (ER) field of study started to gain shape. Using this approach, an initial random controller is optimized through several generations. At each generation, a population of candidate solutions is tested and the best performing solutions are mutated, crossed-over and passed on to the next generation. With this method, we get an incrementally better controller at each generation as we let evolution take care of the controller specification.

A common framework for robotic controllers is an artificial neural network (ANN). This approach is inspired by the way the human brain works, with computer models of axons and neurons. One of the main advantages of the ANN framework applied to robotic controllers is the resistance to noise (Jim et al., 1995), introduced for example by the normal imperfections of real-world hardware (sensors). The ANN framework is also a natural fit for robotics, with its layer architecture allowing for a direct mapping of the sensors to the input layer and the actuators to the output

layer. Sensor activation in ANN's are usually represented by a value in specific range, for example [0,1].

Environment perception in robotics is a natural evolution driven by the need to make robots ever more autonomous and intelligent. Different approaches on this subject have been studied over the years, based on voice (Fitzpatrick, 2003), vision (Merino et al., 2006; Spaan, 2010; Spaan et al., 2010) and touch (Le et al., 2010) to perceive the environment.

Investigation on this subject although very sparse in the means of perceiving and acting upon the environment, concerns mostly terrain environments. With the proliferation of devices like drones and the expansion of robotic applications, it's important to explore different environments and create solutions that can be applied to multiple scenarios. In particular, this work will focus on simulating conditions characteristic of terrain, aerial and marine environments and the challenges that arise in both developing cooperative active perception capabilities for swarms that are scalable to multiple environments and the new challenges introduced by each of the environments' singularities.

In the scope of this article, perceiving the environment can be described as the identification of objects, its features and further classification. Upon the results of that classification, the robot can act on the environment, changing its state. The perception of each robot is shared with the team-members in the field of sight. This aggregates as a cooperative active perception approach to swarm robotics.

This task is required for complex environments where observations must be verified by several sources for structures that have a much larger scale than the sensors, or that need to be sensed in different wavelengths, or using different types of sensors, thus requiring the contribution of different elements of the swarm each, identifying a specific set of characteristics, to validate the identification. In this case the problem was simplified to sets of different color that had to be observed at the same time and communicated to the peers.

We will focus in a task where a swarm of robots navigates through an environment crossed by unidentified objects. These objects carry a set of features, each of which can only be observed from a different viewpoint. The robots have three goals:

1. Identifying all the features of the objects

2. Catching the objects that fall in a certain category defined by the presence of a specific set of features.

3. Keeping a formation like distribution on the environment, simulating a patrolling behavior inside the arena.

Although collective object identification is not a novel issue, the introduction of marine and aerial singularities and the expectation of creating an environment independent solution was not approached with depth in previous studies and can have relevant applications, from marine surveillance operations to aerial forest fires detection.

In summary, the key objectives are:

1. Develop a cooperative active perception approach that is scalable to different types of environments and its singularities.

2. The demonstration of the approach successfully working on a simulation environment with known real-world transferability (Duarte et al., 2012).

The main contributions are:

1. The assessment of the results in evolving a solution to a new learning task, suited to test cooperative perception problems.

2. The evaluation of techniques to evolve more robust solutions that adapt to different environments.

2 RELATED WORK

Sensing the environment is one of the key features to enable a fully autonomous behavior. To successfully develop a controller with these capabilities, several problems need to be considered, in multiple areas: environment perception, object recognition and computer vision.

In this section, we start with an overview of ER, the technique that will be used in the synthesis of the robotic controllers developed throughout this study and we review various approaches studied for solving the cooperative active perception challenges in swarm robotics for autonomous robots.

2.1 Evolutionary Robotics

Evolutionary computation is a sub-field of artificial intelligence in which evolutionary algorithms (EAs) are used. These algorithms are inspired on biological mechanisms, following the same principles as the natural evolution described by Darwin. The fitness function plays one of the most important roles in the evolution, defining the balance of the objectives to be reached in order to get the most adequate solution after a couple generations.

ANNs are the most common framework of ER controllers. This approach is inspired by the way the human brain processes information, like biological

neurons (McCulloch and Pitts, 1943), with nervous activities, neural events and relations being described in terms of propositional logic.

A typical neural network includes five components: (i) the input layer, (ii) the hidden layer, (iii) the output layer, (iv) the weighted connections between each of the previous components and (v) the activation function that converts the input to the output in each of the nodes (neurons). The weighted connections as well as the activation function for the neurons are the main parameters that define an abstract ANN framework to solve a concrete problem. When EAs are used, these parameters are obtained via the global optimization methods characteristic of this approach. This process replaces the manual specification of the solution and it is the main advantage of using this method.

Early approaches were often based on a specific type of ANN, a discrete time neural network. Continuous-time recurrent neural networks (CTRNN) were later introduced by Joseph Chen in 1998 with appealing results (Chen and Wermter, 1998), filling the gap of the discrete time neural network's lack of temporal dynamics, like short term memory.

ER comes as a natural concretization of EAs to synthesize robotic controllers. These methodologies started emerging in the 1990's (Lewis et al., 1992; Cliff et al., 1993). Even when the fitness function didn't imply certain attributes, the evolution developed those capabilities to solve the task. The authors consider the results sufficiently promising of future success in the area.

Although the approach has proven successfully in evolving creative solutions for simple behaviors like foraging, formation, aggregation, etc, one of the biggest challenges in the area is scaling up the approach to more complex tasks, mainly due to the bootstrapping problem, where the goal is so hard/distant that all individuals in the first generation perform equally bad, causing a slow start of the evolution process. Transferring the robotic controllers from simulation to real environments (crossing the reality gap) is another big challenge, with proposed solutions based on sensors, noise and real-world error estimation (Angelo Cangelosi, Domenico Parisi, 1994; Jakobi et al., 1995; Hartland and Bredèche, 2006).

In 2007, M. Eaton presents one of the first application of EAs to develop complex moving patterns of a humanoid robot (Eaton, 2007) and successfully transfers the solution to real hardware.

Miguel Duarte conducted a study (Duarte et al., 2012) that introduced a novel methodology for developing controllers for complex tasks: recursively splitting them into simpler tasks until these are sim-

ple enough to be evolved; controllers to manage the activation of these tasks are also evolved. Then, a tree-like composition of simple tasks and its activation controllers make up the solution for the initial complex task.

2.2 Cooperative Active Perception

As referred by Paul Fitzpatrick in (Fitzpatrick, 2003), it is difficult to achieve robust machine perception, but doing so is the key to intelligent behavior. The author also defends an active perception approach, as figure/ground separation is difficult for computer vision. This author conducted studies using active vision and active sensing for object segmentation, object recognition and orientation sensitivity.

In 2006, Luís Merino (Merino et al., 2006) used a cooperative perception system for GPS-equipped Unmanned Aerial Vehicles (UAV)'s to detect forest fires, where active vision played the most important role. A statistical framework is used to reduce the uncertainty of the global objective (the fire position) taking into account each team-member distinct sensor readings and their uncertainty. This approach provides a way to exploit complementarities of different UAV with different attributes and sensors.

The foundation of all this process is profoundly linked to a robust perception, as such, correctly identifying objects. As stated by Q. V. Le in (Le et al., 2010), angles in which objects can be viewed are the main variable to increase likeliness of identification. This study produces great results in object identification as the robot is capable of observing the object in many angles until certainty is reached, and was proven to be better than passive observation and random manipulation.

To drive the robot's decision making based on an incomplete and noisy perception is another challenge described in 2010 by Matthijs T.J. Spaan in (Spaan, 2010) and (Spaan et al., 2010). The authors propose a Partially Observable Markov Decision Process (POMDP) to develop an integrated decision-theoretic approach of cooperative active perception, as POMDPs *"offer a strong mathematical framework for sequential decision making under uncertainty, explicitly modeling the imperfect sensing and actuation capabilities of the overall system."*. Later in 2014, the authors introduced a new type of POMDP, POMDP-IR (Information Reward), that extends the solution with actions that return information rewards (Spaan et al., 2014).

Another robot control approach for a perception-driven swarm is presented by Aamir Ahmad in 2013 (Ahmad et al., 2013), where the author proposed and

implemented a method for a perception-driven multirobot formation control, with a weighted summed term cost function to control multiple objectives. This study was successful in demonstrating that the authors' approach enables a team of homogeneous robots to minimize the uncertainty of a tracked object while satisfying other criteria such as keeping a formation. The approach consists in integrating two main modules, a controller and an estimator.

Seong-Woo Kim states that fusing data from remote sensors has various challenges (Kim et al., 2015). The author focuses on the map merging problem and sensor multimodality between swarm members to successfully extend perception range beyond that of each member's sensors. Compared with cooperative driving without perception sharing, his approach was proven better at assisting driving decisions in complex traffic situations. The author proposes triangulation and map reckoning to get the relative pose of the nodes allowing the information to be properly fused. The approach assumes no common coordinate system making it more robust.

In 2015, Tiago Rodrigues addressed the sensor sharing challenges as well. In (Rodrigues et al., 2015) the author proposes local communication to share sensor information between neighbors to overcome constraints of each member's local sensors. Triangulation is used to georeference the tracked object. The proposed approach is transparent to the controller, working as a collective sensor. This scenario was able to achieve a much better performance than classic local sensors.

These techniques present a diverse contribution in terms of the robotic controllers used, and the sensing and actuating capabilities. In most of the cited work, active vision played the central role of the approach (Fitzpatrick, 2003; Merino et al., 2006; Le et al., 2010). In (Merino et al., 2006), a statistical framework is used in the controllers to estimate the target position, and perception with heterogeneous teams is tested. In (Spaan et al., 2010; Spaan, 2010; Spaan et al., 2014), POMDP's were used to model decision making under uncertainty (good for noisy perceptions). The control of multiple objectives in a robotic solution is presented in (Ahmad et al., 2013). Fusing data sensed between multiple nodes also poses challenges studied in (Kim et al., 2015), and (Rodrigues et al., 2015) presents a shared sensor solution to the same problem.

The studies presented above develop and test perception solutions centered in the linear terrain environment, and the development of cooperative active perception systems using ER was not approached with depth. The work presented in this study differs in proposing a generic solution scalable to environments with different characteristics and overcoming the challenges of the environments' singularities, using ER techniques.

3 METHODOLOGY

We will now proceed to describe an approach for a swarm robotics control system capable of collectively identifying objects and making decisions based on the identification. It's a common approach in robotic perception to unfold the identification of objects as the identification of specific features that build to a known object or class of objects (Fitzpatrick, 2003; Le et al., 2010). Our approach follows that direction: the identification of an object is completed when all its key features are seen by at least one of the robots in the team. Those features can be sensed: (i) directly by each team member using its local sensor and (ii) indirectly through the shared sensor that allows each robot to sense object features being seen by the teammates in sight. From the controller's point of view, there is no distinction between the local and the shared sensing of a feature.

In this work, the objects and its features serve as a conceptual representation of any given category of object and its features, respectively.

We'll use a task in which a team of robots must collectively identify a set of objects that pass by, and catch the ones that fall into a certain category (have a specific set of features).

For our experiments, we will use JBotEvolver (Duarte et al., 2014) , a Java-based open-source neuroevolution framework and versatile simulation platform for education and research-driven experiments in ER.

3.1 Experimental Setup

To conduct our experiments, 8 circular robots with a radius of 5 cm are placed in a 4x4 m bounded environment. The unidentified objects have a 10 cm radius (twice the size of the robots), are generated in intervals of 500 time steps (50 seconds) and can appear from any side of the arena, moving to the opposite side. In 30% of cases, two objects will be on the arena at the same time, increasing the identification complexity; in the remainder 70% of cases only one object is inside the arena at the same time. The initial position of the object is randomly assigned when only one object is on the arena at a time and fixed on the bottom and top or left and right portions of the arena when two objects are on the arena at the same time.

Having two objects inside the arena at the same time should force the robots to separate in groups to proceed with the identification. Object speed is variable, assigned to each object at the moment of creation and corresponding to a random speed between 0.15 and 0.35 cm/s, drawn from a uniform distribution.

Each object carries 4 features distributed around the 4 quadrants of the object's circular perimeter. In the scope of this study, object features are represented by colors. To simulate the complexity associated with large objects identification (objects bigger than robots) and scale the approach to multiple object sizes, each robot can only see one feature at a time. With this limitation, cooperation is needed to sense all the features and proceed with the identification. The key is for the robots to position themselves around an object so that each one is situated in a vantage point that enables it to see one feature directly through its local sensor and all the others indirectly, through the shared sensor that receives the perceptions from nearby teammates. An object is considered identified if all the features are observed by a robot, for 10 consecutive time steps.

The object features are contained in a predefined set of 8 features (4 enemy features and 4 friend features), unknown by the robots. While enemy objects always have 4 enemy features, friend objects can have a mix of friend and enemy features (up to a max of 2 enemy features). This ambiguity serves a more realistic model and forces robots to evolve a more precise identification process. The order, mix and choice of the features are all uniformly distributed random processes that take place at the generation of each object.

An example of the object identification scenario is depicted on Fig. 1Schematics of the simulation environment when identifying an object. Object 0 represents the unidentified object, with f1 to f4 representing its features; robot 0 to robot 3 represent the swarm; grey filled sensors represent the local features sensor of each robot; C represents the communication between each team-member (shared features sensor) and the circular lines represent the field of communication of each robot and its teammates (radius of robot sensor).. Here, each robot is sensing a different feature of the object with its front facing local sensor. All robots are inside of each other's range of communication, thus being able to share the local perception. As a result, the 4 robots are able to identify the object, as each of them knows all the features. They are now able to deduce it's category and decide whether they should catch the object if it's an enemy (any of them can take that action).
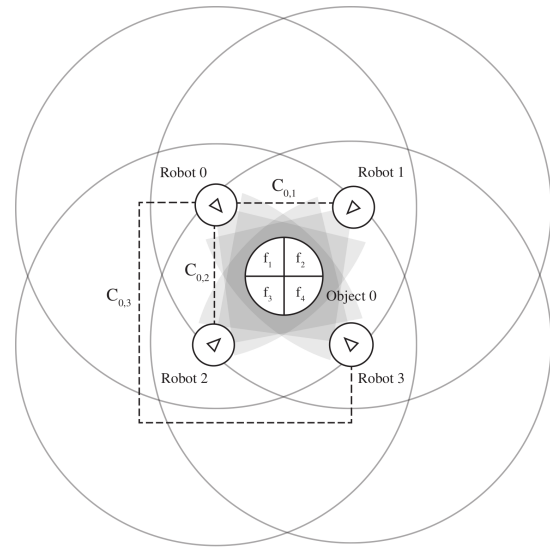

Figure 1: Schematics of the simulation environment when identifying an object. Object 0 represents the unidentified object, with f1 to f4 representing its features; robot 0 to robot 3 represent the swarm; grey filled sensors represent the local features sensor of each robot; C represents the communication between each team-member (shared features sensor) and the circular lines represent the field of communication of each robot and its teammates (radius of robot sensor).

3.2 Controller Architecture

The robotic controller will be obtained using the AI methods introduced in section 2.1 and is driven by a CTRNN. The optimization will be set to maximize a fitness function that measures the solution performance.

The controller architecture is composed of 2 actuators and 5 sensors. The information from the environment perceived by the robot through its sensor is represented by a [0,1] value and mapped to the neural network inputs. A hidden neuron layer is also used, with 5 hidden neurons. The neurons in this layer are connected to each other and to themselves, maintaining a state (this allows for short term memory). The output layer of the ANN is connected to the robot's actuators.

An array of wall, robot, distances, features and team-mate density sensors were chosen. Together, they provide all the necessary information to successfully solve the proposed task. All the sensors, actuators and corresponding ANN inputs and outputs are described in Table 1Controller Architecture: Robot Sensors and Actuators and corresponding ANN Inputs and Outputs..

The following equation describes the network behaviour:

Table 1: Controller Architecture: Robot Sensors and Actuators and corresponding ANN Inputs and Outputs.

Sensor	ANN Inputs
i) Wall Sensor Reading in range [0,1] depending on distance to closest wall	4 (total of 4 sensors around the robot each with 90° aperture)
ii) Robot Sensor Reading in range [0,1] depending on distance to closest robot	4 (total of 4 sensors around the robot each with 90° aperture)
iii) Object Distance Sensor Reading in range [0,1] depending on distance to closest object	4 (total of 4 sensors around the robot each with 90° aperture)
iv) Object Features Shared Sensor Binary readings corresponding to the feature in sight for the closest object	8 (local) + 8 x $N_{close\ robots}$ (shared) (2 local sensors arranged like eyes, with 35° aperture and 10° between the eyes)
v) Robot Density Sensor Reading corresponding to the percentage of robots in sight according to total	1 (1 sensor)
Actuator	ANN Output
i) Differential Drive Actuator Output in range [0,1] depending on speed	2 (left and right)
ii) Object Catch Actuator Binary output to catch an object	1 (catches closest object at max distance of 0.1 m)

$$\tau_i \frac{dH_i}{dt} = -H_i + \sum_{j=1}^{in} \omega_{ji} I_i + \sum_{k=1}^{hidden} \omega_{ki} Z(H_k + \beta_k) \quad (1)$$

with

$$Z(x) = (1 + e^{-x})^{-1} \quad (2)$$

where τ_i represents the decay constant, H_i the neuron state and ω_{ji} the strength of the synaptic connection between neurons j and i (the weighted connections). β represents the bias and $Z(x)$ is the sigmoid function (equation 2). *in* represents the total number of inputs and *hidden* the total number of hidden nodes (5 were used). β, τ and ω_{ji} compose the genome that encodes the controller behavior, and are the parameters randomly initialized at the first generation and optimized throughout the evolutionary process, where $\beta \in [-10, 10]$, $\tau \in [0.1, 32]$ and $w_{ji} \in [-10, 10]$. Integrations follow the forward Euler method with an in-

tegration step size of 0.2 and cell potentials set to 0 at network initialization.

The sensors follow the configuration depicted on Fig. 2Robot sensors representation. Sensors i), ii) and iii) are placed all around the perimeter of the robot and sensor iv) consists in 2 front facing sensors with an eye-like distribution, for a more realistic approach since the perception is based on vision. This also allows the robot to sense the path to reach the object (due to the sensors overlapping at the center).

To catch the objects, robots have a binary actuator. When active, the closest object is caught by the robot if situated at a maximum distance of 0.1 m.
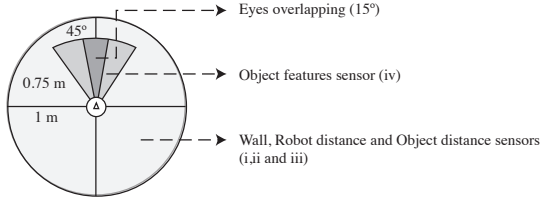
Figure 2: Robot sensors representation. 4 sensors with 90° opening angle for sensors i), ii) and iii) and 2 eyes-like sensor with 45° opening angle and 15° of overlapping for sensor iv).

3.3 Fitness Function and Evolutionary Process

To obtain the controller, the evolutionary process was conducted 10 times (evolutionary runs) during 2000 generations. Each generation is composed of 100 individuals, each corresponding to a genome that encodes an ANN. To select the best individuals in a generation, the considered fitness is the average of 30 tests. Each sample is tested during 5000 time steps (500 seconds). For the test, every robot in the swarm has the same genome. After each individual is evaluated, the top 5 are included in the next generation and used to create the remanding 95 individuals of the population: each one of the top individuals generates 19 new individuals by applying gaussian noise to each genome with a probability of 10%.

The fitness function is very simple: it rewards robots for identifying and catching enemies and penalizes them for catching friends. A formation component was added to the fitness function, to stimulate the robots to evolve a patrolling behavior and spread out inside the arena, maintaining a known distance to each other. The evolution is set to optimize this fitness function, described in equation 3:

$$F_i = \alpha_i + \beta_i \tag{3}$$

with

$$\alpha_i = \sum_{n=0}^{timesteps} 10^{-2}, \text{if ADN} \in [S_r - \tfrac{S_r}{10}, S_r + \tfrac{S_r}{10}] \tag{4}$$

or

$$\alpha_i = \sum_{n=0}^{timesteps} -|ADN - S_r| \times 10^{-2}, otherwise \tag{5}$$

and

$$\beta_i = \frac{Enemy_{identified}}{5 \times 10^{-3}} + \frac{Enemy_{caught}}{10^{-3}} - \frac{Friends_{caught}}{2 \times 10^{-3}} - \frac{Unidentified_{caught}}{10^{-3}} \tag{6}$$

α_i and β_i correspond to the formation component of the fitness function and the object identification component, respectively. *ADN* (Average Distance to Nearest) is the average distance of the robots to its closest team-mate, S_r is the robot teammates sensor radius. $Enemy_{identified}$ is the total number of enemy objects that were identified during the test, $Enemy_{caught}$ corresponds to the total number of enemy objects caught. $Friends_{caught}$ and $Unidentified_{caught}$ corresponds to the total number of friends / inoffensive objects caught, respectively. The formation component rewards the robots for keeping a distance between each other that corresponds to the radius of their teammates sensor (S_r) with an error margin of $\frac{S_r}{10}$. This allows them to disperse around the environment in search for objects while keeping a known distance to their teammates.

4 MULTIPLE ENVIRONMENTS

The global contribution of this work is not only to present a novel cooperative active perception solution using EAs, but also to fill a gap in the current state of the art: evaluate the possibility of evolving generic solutions, adaptable to environments with multiple characteristics and its singularities.

In the real-world, external factors heavily influence the swarm performance. In this section, we will model different environments, mainly governed by external conditions that influence the swarm performance. Three main classes of environments will be considered: (i) terrain, (ii) marine and (iii) aerial.

On a terrain environment, we simulate visual and navigational obstacles present on terrain scenarios. While terrain irregularities can be handled by the robotic driver and thus don't need to be handled by the controller, accessibility issues like obstacles or object occlusion will benefit from an optimized behavior to solve the task in these conditions. In our model, we included a set of rectangular opaque obstacles distributed around the environment.

The marine environment can help develop swarms capable of running patrolling and exploration marine tasks. Our model of the marine environment is based on previous studies that successfully obtained controllers capable of crossing the reality gap in a marine environment (Duarte et al., 2016) and is centered around two main characteristics: (i) a constant dragging current and (ii) robots movement inertia. Each robot has two marine propellers (left and right) that are controlled by the differential drive actuator.

Regarding the aerial environment, several studies (Pflimlin et al., 2004; Cheviron et al., 2009; Leonard

et al., 2012) address some of the challenges of controlling an Unmanned Aerial Vehicle (UAV): maneuverability, wing gusts and other aerodynamic efforts. In (Cheviron et al., 2009), the authors study the influence of wind gusts on the system concluding that it is a crucial problem for real-world outdoor applications, especially on an urban environment. We will base our model of the aerial environment in the simulation of: (i) constant wind and (ii) intermittent wind gusts, as these seem the most relevant challenges. This environment can be used to achieve controllers capable of drone obstacle avoidance and object detection.

All the agent's solutions are built upon the solution presented in section 3.1. A description of each of the environments and it's singularities is summed in table 2Multiple Environments: Description of each environment and it's singularities..

Obstacles width and height, sea current and wind magnitudes, gust duration and whether a gust is present or not are all random values drawn from a uniform distribution. The intervals used were designed, by trial and error, to include different problems for the agents to solve in each environment, although all on a similar difficulty level.

5 RESULTS AND DISCUSSION

As we place the robots in different settings, the optimization will follow different paths and we obtain different solutions, specifically optimized to the setup the evolutionary process was conducted within. The evolutionary process described in section 3.3 was conducted in four main setup categories: (i) in each of the 3 environments described, (ii) in an ideal setup (described in section 3.1), (iii) in a noisy environment and (iv) in a hybrid scenario - consists in each sample being conducted in a different environment (terrain, marine or aerial). A total of 60 evolutionary processes were conducted, taking 29 days to complete on a computer grid with an average availability of 75 workers.

Conducting the evolutionary process in each of the 3 environments gives us a benchmark for the target controller behavior in each environment. This way, we will obtain controllers specifically optimized for each environment. If we find controllers obtained via other methods (ie. noisy or hybrid) to perform as good as the environment specific controllers, the most generic solution will be validated.

The noisy environment was introduced as noise can be seen as an abstract and multi-purpose way of generating a more robust solution. Introducing noise on the ANN Inputs during the evolutionary process is one of the known ways of creating a solution that

is able to cope with slightly different conditions than the ideal environments usually used during training, thus boosting the ability to cross the reality gap. All sensors are affected by the noise with a fixed offset of [-0.1,0.1] and random noise [0.1,0.1], for each reading as suggested in (Romano et al., 2016). For object features, a 10% probability of having each binary reading reversed is used, a value equivalent to the previous. Offset, noise values and binary state reversions are random processes drawn from an uniform distribution.

In the hybrid scenario, 1/3 of the samples are conducted in the terrain, marine and aerial environments.

Controllers will be tested not only in the environment they were evolved in but also in all the others. The evolutionary process was conducted to optimize the fitness function set in Eq. 3 with the configuration detailed in Table 1Controller Architecture: Robot Sensors and Actuators and corresponding ANN Inputs and Outputs.. The tests are done to the best controller resulting from the evolution, with an average of 100 samples during 10000 time steps.

Results for each evolution are condensed in Table 3Each evolution tested in each environment. in terms of: fitness, percentage of friends and enemies identified and percentage of enemies and friends caught. In Figure 3All controllers tested in all scenarios we can analyse the fitness dispersion of each controller, tested on each scenario.

Although environment specific evolution provided good results, it was not always the best option. The terrain environment is an example: the ideal and marine evolved controllers had better performance when tested on the terrain environment, with an average fitness of 2576 ± 1340 and 2191 ± 1261, respectively, while the terrain evolved controller scored a fitness of 2158 ± 1433 (15% lower). Although the margin is small, it stands out the fact that the terrain evolved scenario was not the best fit to solve the task in the environment it was trained in, possibly due to the complexity of the scenario preventing the evolution from extracting the object identification and catching as well as it did on the ideal environment. The characteristics of the terrain environment conducted the evolution to a behavior in which the swarm separates in small search groups strategically placed in spaces confined by the obstacles.

When tested on the noisy environment, all controllers failed to solve the task. Although the noise magnitude used in these experiments gave us good results in previous studies (Romano et al., 2016), it appears to be destructive in this scenario. In previous studies, we used a simple aggregation and formation task with identical noise applied. The controller we

Table 2: Multiple Environments: Description of each environment and it's singularities.

Environment	Singularities
Terrain Environment	i) [0,7] obstacles inside the arena, width and height between [25,65] cm
Marine Environment	i) $F_{\text{current}} \in [-0.1, 0.1]$ cm/s for each axis fixed throughout each sample (constant current)
	ii) propellers movement inertia, with a maximum increment of 0.1m/s for each timestep,
	for each propeller.
Aerial Environment	i) $F_{\text{current}} \in [-0.1, 0.1]$ cm/s for each axis fixed throughout each sample (constant wind)
	ii) $F_{\text{gust}} \in [-2, 2]$ cm/s for each axis fixed fixed throughout the gust period. $Gust_{\text{period}} \in [0, 20]$
	seconds, sorted at the beginning of each wind gust.
	Wind gusts are intermittent, sorted between silent and windy periods.

present in this work shares many of the same sensors and actuators as the solution on the previous study, the biggest difference being the shared features sensor. While the search and identification portion of the behavior seems correct, the categorization was the main variable to fail in the controller (that caught both enemies and friends), leading us to conclude the shared features sensor was the bottleneck that caused the noisy evolved controller to fail, being the component less prone to noise.

The marine environment is the environment with the biggest discrepancy between the environment specific evolution performance and the remaining, with the environment specific controller scoring an average fitness of 3473 ± 1549. Hybrid evolved controller on this environment scored a lower fitness of 2242 ± 1120. The ideal, terrain and aerial evolved controller scored the lowest fitness by a big margin: 843 ± 1518, 694 ± 1415 and 768 ± 901, respectively. The robot movement inertia is the main difference in this environment. This results shows us that although the adaption to this characteristic is needed (low fitness on the ideal evolution), the adaption is not hard for the evolution to handle (high fitness in the environment specific and hybrid controllers). Direct observation of the behavior presents no visible differences to the remaining solutions.

The aerial environment presented the lowest global fitness values among the three environments. With no clear performance distinction from the environment specific solution, we conclude that the evolution was not able to generate a controller that compensates for the wind gusts. Observing the behavior, we notice that when the wind gusts appear, the robots

lose control of the object being identified. Controllers evolved in the aerial environment revealed a tendency to always keep close together (behavior found on 90% of the evolutionary runs). This tendency was not observed in the remaining scenarios and represents a specific path the aerial evolution followed, possibly keeping teammates close to use them as a reference to acquire awareness of when the wind gusts drag the robots out of their position.

We noted that on all environments, the hybrid controller performance revealed to be on par with the environment specific results in terms of fitness. To further analyze these results, the differences between the environment specific controllers and the hybrid controller are condensed on Table 4Environment specific controllers compared to the hybrid evolved controller in each scenario., for: (i) "enemies identified ratio", (ii) "enemies caught ratio" and (iii) "friends and unidentified objects caught ratio".

We notice that the differences between the two approaches range from a positive performance of [0,2]% for the hybrid controller in the terrain and marine environment and a slight degradation of performance of [1,8]% in the aerial environment.

The hybrid controller reveals to be equivalent to the environment specific controllers in the terrain and marine environments, and worse on the aerial environment. The worse performance on the aerial environment is common to most of the experiments, possibly linked to the overall complexity of this environment. Still, the differences found between these are of small magnitude. In terms of observable behavior, there are no visible differences as both solve the task in the same manner. We can state that the perfor-

Table 3: Each evolution tested in each environment.

Testing scenario	Fitness ± Stdev	Enemies identified (%)	Enemies caught (%)	Friends caught (%)	Unidentified caught (%)
Terrain environment evolution					
Ideal Environment	2874 ± 1922	54%	42%	1%	5%
Noisy Environment	-529 ± 1713	5%	0%	0%	4%
Terrain Environment	2158 ± 1433	41%	32%	1%	3%
Marine Environment	694 ± 1415	25%	11%	0%	3%
Aerial Environment	752 ± 801	22%	13%	0%	3%
Hybrid Environment	1263 ± 1056	31%	20%	1%	3%
Marine environment evolution					
Ideal Environment	3067 ± 1668	56%	46%	2%	7%
Noisy Environment	-492 ± 1600	4%	0%	0%	4%
Terrain Environment	2191 ± 1261	43%	34%	2%	5%
Marine Environment	3473 ± 1549	59%	51%	2%	7%
Aerial Environment	737 ± 684	24%	13%	0%	4%
Hybrid Environment	2242 ± 1120	42%	34%	1%	5%
Aerial environment evolution					
Ideal Environment	2385 ± 1026	48%	40%	6%	7%
Noisy Environment	92 ± 210	15%	2%	3%	1%
Terrain Environment	1715 ± 799	37%	31%	5%	5%
Marine Environment	768 ± 901	26%	12%	1%	3%
Aerial Environment	1428 ± 618	35%	23%	2%	4%
Hybrid Environment	1334 ± 777	34%	23%	3%	4%
Ideal environment evolution					
Ideal Environment	4239 ± 2573	71%	64%	2%	7%
Noisy Environment	-90 ± 515	6%	0%	0%	1%
Terrain Environment	2576 ± 1340	45%	37%	1%	3%
Marine Environment	843 ± 1518	25%	12%	0%	2%
Aerial Environment	836 ± 748	22%	13%	0%	3%
Hybrid Environment	1557 ± 1112	33%	23%	0%	3%
Noisy environment evolution					
Ideal Environment	2120 ± 428	59%	47%	42%	4%
Noisy Environment	2186 ± 319	58%	48%	43%	4%
Terrain Environment	1526 ± 209	45%	36%	32%	4%
Marine Environment	191 ± 714	25%	15%	13%	6%
Aerial Environment	443 ± 193	22%	10%	9%	2%
Hybrid Environment	809 ± 388	31%	20%	18%	3%
Hybrid environment evolution					
Ideal Environment	3031 ± 1292	54%	45%	4%	6%
Noisy Environment	157 ± 220	16%	0%	0%	0%
Terrain Environment	2108 ± 951	41%	34%	3%	4%
Marine Environment	2553 ± 1176	48%	38%	2%	6%
Aerial Environment	1057 ± 483	27%	16%	1%	3%
Hybrid Environment	2004 ± 881	40%	31%	2%	4%

Table 4: Environment specific controllers compared to the hybrid evolved controller in each scenario.

Environment	Terrain Environment		Marine Environment		Aerial Environment	
Controller	Env. specific	Hybrid (± diff)	Env. specific	Hybrid (± diff)	Env. specific	Hybrid (± diff)
Enemies identified (%)	41%	41% (0%)	43%	48% (+5%)	35%	27% (-8%)
Enemies caught (%)	32%	34% (+2%)	36%	38% (+2%)	23%	16% (-7%)
Friends caught (%)	1%	3% (+2%)	2%	2% (0%)	2%	1% (-1%)
Unidentified caught (%)	3%	4% (+1%)	5%	6% (+1%)	4%	3% (-1%)

mance for the hybrid controller on the terrain, marine and aerial environments is similar to the one obtained by evolving specific controllers, differing only by a small negligible margin with no clear performance impact.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we proposed a novel approach for swarm robotics environment perception. This approach is different from the remaining state of the art
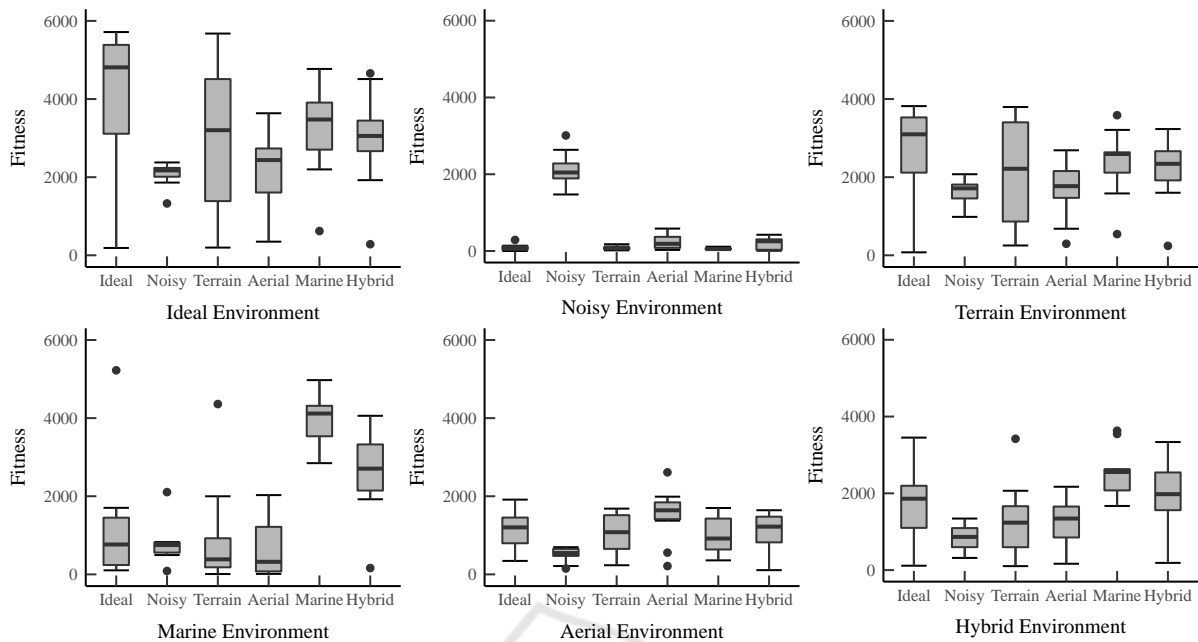
Figure 3: All controllers tested in all scenarios.

for two main reasons: (i) the controller is obtained using EAs and (ii) the study is focused on scaling the approach to multiple environments.

We conducted the study in a simulation scenario, with unidentified objects appearing from any side of the screen moving to the opposite side, with the possibility of having two objects on screen at the same time. The evolved behavior consists in performing a dispersed search around the arena, getting closer to the objects when an enemy feature is detected. When robots gather around the object, one of them catches it. Attention given to friend features is lower, so robots didn't gather around the friend objects most times, nor caught them.

Besides the ideal environment, we also modeled: (i) a terrain environment based on obstacles randomly placed around the environment, (ii) a marine environment with constant currents and inertia in the robots' movements and (iii) an aerial environment with a constant current and wind gusts. Also, we selected 2 main scenarios that are known to evolve more robust behaviors: (i) noisy evolution and (ii) a hybrid evolution in the multiple scenarios. These were compared to the ideal evolution scenario.

When observing the evolved behaviors, two main categories can be extracted: in the first, the robots evolved a behavior in which the team performs a dispersed search around the arena and then aggregates around the object to proceed with the identification; in the second, the robots follow each other in circu-

lar paths around the environment once again aggregating towards the object to identify. The identification process followed very similar behavior in all experiments: circumnavigating the object while front-facing it until the identification is complete. Specialization was also observed on the environment-specific evolutions: in the terrain evolved controller the swarm had a tendency to separate in groups and search inside the areas confined by the obstacles.

The noisy evolution not only failed to evolve a more robust and scalable solution, but also failed to solve the task at all. The noise magnitude that was adequate for similar tasks (Romano et al., 2016) revealed to be destructive for this task. The global objective of this work was to test and compare several ways of developing a controller. The controller should be capable of collectively identifying and categorizing a set of objects and act upon multiple types of environments based on the categorization. This objective was successfully completed as we demonstrated how EAs could synthesize a controller capable of solving this task. We have also tested the flexibility of a controller trained in multiple environments: the hybrid solution. Although environment-specific controllers globally outperformed the hybrid controller in the respective environment, the difference between the two was small enough to state that both controllers are equally capable of solving the task.

Future work could start with the scaling of the approach using 3D models of the environments. This

would allow for a more realistic simulation with major impact specifically on the aerial environment, where the 2D representation used in this work is a major simplification. Another necessary step is the deployment of the solution to real robots and real environments, optimization and study of the challenges associated with it. The biggest difficulty for the controller appeared to be on the aerial environment, specifically the wind gusts, that the controller had difficulty in compensating. Future work could also reside in optimizing this controller for better results in the different environments. For example, giving the controller access to a sensor that detects wind gusts could help the robot compensate them and boost the performance on the aerial environment.

ACKNOWLEDGMENTS

REFERENCES

Ahmad, A., Nascimento, T., Conceicao, A. G. S., Moreira, A. P., and Lima, P. (2013). Perception-driven multi-robot formation control. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1851–1856.

Angelo Cangelosi, Domenico Parisi (1994). The touch sensitive behavior of caenorhabditis elegans: A simulation approach using neural networks. Technical Report May, Institute of Psychology C.N.R. - Rome.

Chen, J. and Wermter, S. (1998). Continuous Time Recurrent Neural Networks for Grammatical Induction. In *International Conference on Artificial Neural Networks, 1998*, pages 381–386.

Cheviron, T., Plestan, F., and Chriette, A. (2009). A robust guidance and control scheme of an autonomous scale helicopter in presence of wind gusts. *International Journal of Control*, 82(12):2206–2220.

Cliff, D., Husbands, P., and Harvey, I. (1993). Evolving visually guided robots. In *Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB)*, pages 374–383.

Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S. M., and Christensen, A. L. (2016). Evolution of collective behaviors for a real swarm of aquatic surface robots. *PLoS ONE*, 11(3):1–25.

Duarte, M., Oliveira, S., and Christensen, A. L. (2012). Hierarchical evolution of robotic controllers for complex tasks. In *2012 IEEE International Conference on Development and Learning and Epigenetic Robotics, ICDL 2012*.

Duarte, M., Silva, F., Rodrigues, T., Oliveira, S. M., and Christensen, A. L. (2014). {JBotEvolver}: A versatile simulation platform for evolutionary robotics. In *Proceedings of the International Conference on the Synthesis & Simulation of Living Systems (ALIFE)*, pages 210–211.

Eaton, M. (2007). Evolutionary humanoid robotics: past, present and future. *Lecture Notes in Computer Science*, 4850:42.

Fitzpatrick, P. M. (2003). Perception and perspective in robotics. In *Proceedings of the 25th Annual Conference of the Cognitive Science Society*.

Hartland, C. and Bredèche, N. (2006). Evolutionary robotics, anticipation and the reality gap. In *2006 IEEE International Conference on Robotics and Biomimetics, ROBIO 2006*, pages 1640–1645.

Jakobi, N., Husbands, P., and Harvey, I. (1995). Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics. *Lecture Notes in Computer Science*, 929:704–720.

Jim, K., Giles, C. L., and Horne, B. G. (1995). Effects of Noise on Convergence and Generalization in Recurrent Networks. In *Advances in Neural Information Processing Systems (NIPS) 7*, page 649.

Kim, S.-W., Qin, B., Chong, Z. J., Shen, X., Liu, W., Ang, M. H., Frazzoli, E., and Rus, D. (2015). Multivehicle cooperative driving using cooperative perception: Design and experimental validation. *IEEE Transactions on Intelligent Transportation Systems*, 16.

Le, Q. V., Saxena, A., and Ng, A. Y. (2010). Active Perception : Interactive Manipulation for Improving Object Detection. Technical report, Stanford.

Leonard, F., Martini, A., and Abba, G. (2012). Robust nonlinear controls of model-scale helicopters under lateral and vertical wind gusts. In *IEEE Transactions on Control Systems Technology*, pages 154–163.

Lewis, M. A., Fagg, A. H., and Solidum, A. (1992). Genetic Programming Approach to the Construction of a Neural Network for Control of a Walking Robot. In *In IEEE International Conference on Robotics and Automation*, pages 2618–2623.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133.

Merino, L., Caballero, F., Martínez-de Dios, J. R., Ferruz, J., and Ollero, A. (2006). A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires. *Journal of Field Robotics*, 23(3-4):165–184.

Pflimlin, J., Soueres, P., and Hamel, T. (2004). Hovering flight stabilization in wind gusts for ducted fan UAV. *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, 4(January 2005):3491–3496.

Rodrigues, T., Duarte, M., Figueiró, M., Costa, V., Oliveira, S. M., and Christensen, A. L. (2015). Overcoming limited onboard sensing in swarm robotics through local communication. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artifi-*

cial Intelligence and Lecture Notes in Bioinformatics), 9420:201–223.

Romano, P., Nunes, L., Christensen, A. L., Duarte, M., and Oliveira, S. M. (2016). Genome Variations. In Reis, L. P., Moreira, A. P., Lima, P. U., Montano, L., and Muñoz-Martinez, V., editors, *Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volume 1*, pages 309–319, Cham. Springer International Publishing.

Spaan, M. T. J. (2010). Cooperative Active Perception using POMDPs. *October*, pages 4800–4805.

Spaan, M. T. J., Veiga, T. S., and Lima, P. U. (2010). Active cooperative perception in network robot systems using POMDPs. In *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pages 4800–4805.

Spaan, M. T. J., Veiga, T. S., and Lima, P. U. (2014). Decision-theoretic planning under uncertainty with information rewards for active cooperative perception. *Autonomous Agents and Multi-Agent Systems*, 29(6):1157–1185.