Optimising Autonomous Robot Swarm Parameters for Stable Formation Design

Daniel H. Stolfi
SnT - University of Luxembourg
Esch-Sur-Alzette, Luxembourg
daniel.stolfi@uni.lu

Grégoire Danoy
SnT and the FSTM/DCS - University of Luxembourg
Esch-Sur-Alzette, Luxembourg
gregoire.danoy@uni.lu

ABSTRACT

Autonomous robot swarm systems allow to address many inherent limitations of single robot systems, such as scalability and reliability. As a consequence, these have found their way into numerous applications including in the space and aerospace domains like swarm-based asteroid observation or counter-drone systems. However, achieving stable formations around a point of interest using different number of robots and diverse initial conditions can be challenging. In this article we propose a novel method for autonomous robots swarms self-organisation solely relying on their relative position (angle and distance). This work focuses on an evolutionary optimisation approach to calculate the parameters of the swarm, e.g. inter-robot distance, to achieve a reliable formation under different initial conditions. Experiments are conducted using realistic simulations and considering four case studies. The results observed after testing the optimal configurations on 72 unseen scenarios per case study showed the high robustness of our proposal since the desired formation was always achieved. The ability of self-organise around a point of interest maintaining a predefined fixed distance was also validated using real robots.

CCS CONCEPTS

• **Computer systems organization** → **Evolutionary robotics**; • **Mathematics of computing** → *Combinatorial optimization*; • **Applied computing** → Aerospace;

KEYWORDS

swarm robotics, e-puck2, evolutionary algorithm, argos3 simulator, formation control

ACM Reference Format:
Daniel H. Stolfi and Grégoire Danoy. 2022. Optimising Autonomous Robot Swarm Parameters for Stable Formation Design. In *Genetic and Evolutionary Computation Conference (GECCO '22), July 9–13, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3512290.3528709

1 INTRODUCTION

Robot formation as a part of robotics, studies the coordination of a swarm of mobile robots to arrange in a predefined shape [6, 12]. The resulting formation is then used for performing collective tasks, usually in an autonomous way. Applications range from surveillance [3] and synchronisation of spacecrafts [5] to salvage missions [4], caging and grasping [16], and localisation and mapping [22]. These formation proposals have to overcome several problems to be considered a viable solution. Mainly, the initial position of the drones and the path planning from these positions to the final location have to be calculated. However, having predefined final positions restricts the adaptability of the formation to real changing situations and the initial positions are not always known in advance.

We propose a distance based approach where the optimal distances between robots are calculated in order to achieve the final desired formation, surrounding a central point where the target is located. We also take into account the distance of the swarm to the target and the final coverage achieved, i.e. the area of the target seen from the robots. Possible applications are asteroid observation and escorting a rogue drone out of a restricted area, although others are also possible providing the number of robots is enough and the desired formation radius is achievable. To test our proposal we have defined four case studies comprising swarms of three, five, ten, and fifteen robots. We have identified the system's parameters and used a genetic algorithm to optimise them in order to achieve proper formations in all the scenarios tested.

The remainder of this paper is organised as follows. In the next section, we review the state of the art related to our proposal. In Section 3 our approach is presented. The problem is defined in Section 4 where the optimisation algorithm and case studies are also described. The experimental results are in Section 5. And Section 6 brings discussion and future work.

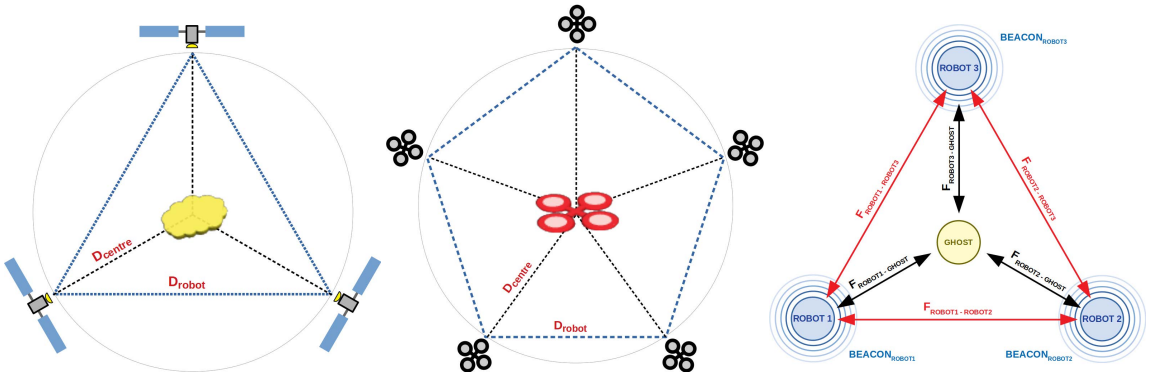
2 RELATED WORK

In this section we review some research works related to distributed formation control and motion planning [15] for autonomous robots. We are interested in distributed/decentralised formation approaches in which robots are not centrally controlled, but autonomously self-organised by using limited local information. This make systems more robust against failures as the final goal is achieved by the collaboration of the individual members of the swarm.

A distributed method for formation control is presented in [1]. Using a restricted communication and vision range, the aerial robots in the swarm share information between them to coordinate and obtain a consensus to navigate in a dynamic environment. The proposed motion planning algorithm is based on calculating the convex hull of the robots' positions and achieving the optimal formation while avoiding obstacles. The results are obtained using Monte Carlo simulation experiments and four quad-rotors. In our proposal we obtain different robot formations since our robots are

GECCO '22, July 9–13, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9237-2/22/07.
https://doi.org/10.1145/3512290.3528709



(a) Three Probes performing asteroid observation (b) Five UAVs surrounding a rogue drone (c) Schema of forces and beacons for three robots

Figure 1: Formation of robots performing asteroid observation (a) and surrounding a rogue drone (b). A detailed schema of forces and beacons for a three-robot example is given in (c).

placed at the vertices of the polygons. Our formations are achieved using attracting and repelling forces optimised by an evolutionary algorithm.

In [18] a team of autonomous mobile robots are controlled using non-linear attractor dynamics. The authors use a formation matrix to define the robot formations for exploration tasks. These formations are achieved using a hierarchical structure of leader-follower robots. Simulations and experiments using real ground robots are shown to demonstrate formation switch and obstacle avoidance are possible without the use of an explicit coordination scheme. Our attractor-repeller forces model differs from this approach in that final formations emerge from the collaboration between swarm members (there is no hierarchy) after optimising their parameters.

A semi-centralised control for multi robot formation is proposed in [25]. The formation problem was modelled as a task distribution problem and solved using the Hungarian method [14]. A leader-followers method was used to get the desired formation and a discussion about the transformation between different formations is also included. Finally, a theorem to optimize the choice of new formation centre was given and simulations were used to test this proposal using MATLAB. There are no leaders and followers in our approach, we propose a different optimisation algorithm to tune the robots' distributed behaviour, and use realistic simulations as well as real robots to obtain and validate our results.

In [13] a deep neural network is designed to model a decentralised formation control policy to map the onboard LIDAR sensor to motor control. Supervised learning is used to train the centralised learning framework and the trained model was deployed on each robot to rely only in local observations without any inter-robot communication. This proposal was evaluated using simulations (VREP) involving three robots. We proposed a different approach based on evolutionary optimisation, use up to 15 robots and test our results using simulations and real robots.

All these works achieve formations using techniques that differ from the one studied in our present article. In our proposal we obtain the robot formation using attracting and repelling forces parametrised using an evolutionary algorithm. We do not take into account external obstacles although collision between swarm members are avoided by the same forces that shape the final desired formation. We use a multi-physics robot simulator to test our proposal on 72 unseen scenarios per case study featuring different initial conditions. To the best of our knowledge, this study involving the optimisation of the swarm parameters for achieving robust formations using an evolutionary technique, followed by the validation using the ARGoS simulator and real world tests using E-Puck2 robots, has not been done before.

3 PROPOSAL

Our proposal for robot formation has the objective of designing and testing an algorithm that can be used for asteroid observation and rogue drone escorting applications (figures 1(a) and 1(b)) among others. Having coverage rates close to 100% (full surface observation) while keeping a safe distance to the central object (target) are desired characteristics of the final formation to reach. Additionally, since working conditions can be harsh, i.e. outer space or conflict zones, autonomy, resilience, flexibility and self-organisation are also required. Fortunately, these characteristics can be reached using swarm intelligence [23], where the collective behaviour of a swarm of robots can be modified through a set of individual parameters to optimise their cooperation and achieve a common goal.

The proposed formation algorithm to be run individually and independently by each robot in the swarm does not require to know the exact position of the other swarm members nor to exchange any data, which makes it more robust against communication issues. It only uses a beacon signal from the other swarm members to obtain the range and bearing (intensity and direction of the incoming radio signal) information and calculate the next moving direction to modify the robot's trajectory accordingly. In order to define a central point for the formation where the object of interest (asteroid, rogue drone, etc.) is located, a number of virtual points (ghosts) are introduced. They are static virtual entities which provide a set of attracting and repelling forces to stabilise the formation around the object to be observed/escorted (Figure 1(c)). Ideally a unique ghost serves as central point, however, as it is discussed in Section 5.2, a

Algorithm 1 Formation Algorithm.

```
1:  function FORMATION(robot, D_centre, D_robot)
2:      a ← 0
3:      for r ∈ ROBOTS ∪ GHOSTS do
4:          if r ≠ robot then
5:              range, bearing ← RangeAndBearing(r)
6:              if r ∈ ROBOTS then
7:                  th ← D_robot                ▷ Distance to other robots
8:              else
9:                  th ← D_centre               ▷ Distance to ghosts
10:             Δ_x ← |th − range| × sin(bearing)        ▷ Intensity
11:             Δ_y ← |th − range| × cos(bearing)
12:             if range < th then
13:                 a ← a − Δ                   ▷ Repelling force
14:             else
15:                 a ← a + Δ                   ▷ Attracting force
16:     return arctan a_y/a_x                   ▷ Next moving direction
```

higher number was needed to increase the influence of the central forces and obtain a stable formation.

Our initial 2D approach is described in Algorithm 1, where the desired distance to the central body and the distance between robots are provided as parameters. In line 2, the value of $\vec{a}$ is initialised to be used later for calculating the resulting vector, after taking into account all the other robots in the swarm as well as the ghosts. For each robot in the swarm and ghosts the range and bearing values are obtained from the radio beacons (lines 3 to 5). In lines 6 to 9 a distance threshold th is selected depending on the nature of r, i.e. whether it is another robot or a ghost. Then, the corresponding $\vec{\Delta}$ is calculated (lines 10 and 11) to modify vector $\vec{a}$ depending of the threshold th (lines 12 to 15). Finally, the next moving direction is returned as an angle (line 16), which depends on the resulting force $\vec{a}$, to be used to modify the current robot's trajectory. After a fixed period of time the Formation algorithm is to be run again for a new trajectory update. Providing the central point is fixed, after a number of iterations the system is expected to be in an equilibrium state where the desired formation shape is achieved and the attracting/repelling forces are mutually cancelled. Under these conditions the Formation algorithm will continue running in each robot to preserve the formation against any environmental change, e.g. the central point has moved or the number of robots in the swarm has changed.

3.1 Formation Parameters

Our most intuitive approach consisted in calculating the geometric distances between robots (D_{robot}) for a given distance to the centre (D_{centre}). Using the equations 1 and 2, the distance D_{robot} can be easily calculated according to the number of robots N and the desired distance D_{centre}.

$$\alpha = 180^{\circ} \times \frac{N-2}{N} \tag{1}$$

$$D_{robot} = 2 \times \cos\frac{\alpha}{2} \times D_{centre} \tag{2}$$

Table 1: Geometric approach ($D_{centre} = 1.00$).

Robots	α	D_{robot}
3	60°	1.732
5	108°	1.176
10	144°	0.618
15	156°	0.416

As we want to test our proposal on swarms made of three, five, ten and fifteen robots, we have calculated the parameters as shown in Table 1. However, after simulating each scenario using the ARGoS simulator [20], we have found that when the number of robots is high (ten and fifteen in our study) there are formations whose final shape is not the expected as shown in Figure 2 and D_{centre} is incorrect.

Our best explanation to this phenomenon is that when having many robots, the desired D_{robot} cannot be achieved since the non-adjacent robots are further away than the others, i.e. the shape's diagonals are longer than the edges. This is not a matter of local minimum but the impossibility for each robot to know whether the others are to be in adjacent vertices, while the formation is being dynamically built. We expect that an equilibrium state can be achieved by using a different parameterisation and possible extra ghosts in order to find the point in which the repelling and attracting forces are cancelled. To address this research question, we propose the optimisation of the system parameters for each case study using a bio-inspired technique: a genetic algorithm. In the following sections we define our formation problem, describe the optimisation algorithm as well as the four proposed case studies, comprising 100 scenarios each.

4 PROBLEM DEFINITION AND OPTIMISATION

In this section we present the problem representation and the evaluation function, describe our optimisation algorithm, and explain the case studies proposed to test our formation system.

4.1 Representation and Evaluation Function

The solution vector representing the system configuration consists of two parameters: the distance between robots (D_{robot}) and the number of ghosts (N_{ghosts}) required for a stable system: $\vec{x} = \{D_{robot}, N_{ghosts}\}$. We have used integer numbers for each parameter to simplify the operators as a precision of two decimal places has proved to be enough in our preliminary tests. Consequently, the range for N_{ghosts} was set to $[1-4]$ while the range for D_{robot} was calculated depending on the number of robots as $[D_{geometric} - 4 \times D_{geometric}]$ where $D_{geometric}$ is the value for D_{robot} calculated by the geometric approach, shown in Table 1. Although two parameters seem to be more appropriated to be optimised using a brute force approach, the complexity of the search space for the combinatorial optimisation associated to our problem makes it suitable for being efficiently solved using a meta-heuristic.

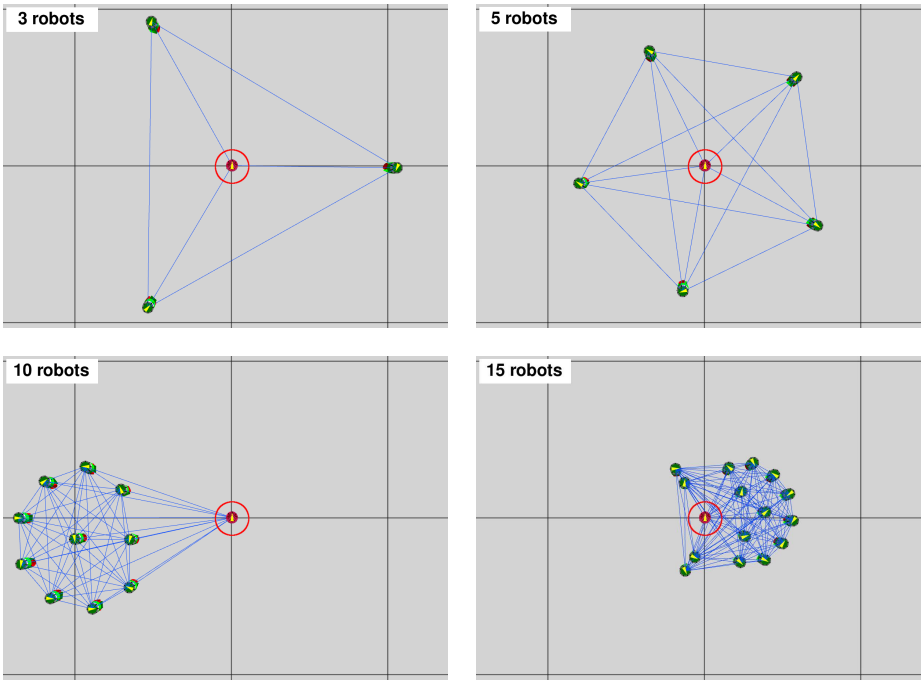


Figure 2: Final formations centred in one ghost when using the geometric distances to configure the robots. It can be seen that 10 and 15 robots achieve undesired equilibrium positions.

The evaluation of each case study is done according to the fitness function $F(\vec{x})$ described by Equation 3. Since we want the optimisation algorithm to calculate a robust configuration, we will evaluate 28 different scenarios ($M = 28$) featuring different initial positions for the robots, and calculate $F(\vec{x})$ using Monte Carlo [17]. Three terms are included in the summation and a penalisation term is also applied to the resulting value. The summation involves the shape of the polygon $P(\vec{x})$ (Equation 4) which is the calculated through the norm of the resulting vector after adding all the final positions of the robots ($P(\vec{x}) = \vec{0}$ for a perfect polygon), and the errors $\varepsilon_{min}(\vec{x})$ and $\varepsilon_{max}(\vec{x})$, (equations 5 and 6) which represent the distances of the closest and furthest robots from the desired D_{centre}.

$$F(\vec{x}) = \frac{1}{M} \sum_{j}^{M} [P(\vec{x}) + \epsilon_{min}(\vec{x}) + \epsilon_{max}(\vec{x})] + \omega(G-1) \quad (3)$$

$$P(\vec{x}) = \| \sum_{i}^{N} \vec{r}_i \| \quad (4)$$

$$\varepsilon_{min}(\vec{x}) = \sum_{i}^{N} |\min[D(\vec{r}_i, \vec{centre})] - D_{centre}| \quad (5)$$

$$\varepsilon_{max}(\vec{x}) = \sum_{i}^{N} |\max[D(\vec{r}_i, \vec{centre})] - D_{centre}| \quad (6)$$

The penalisation term is the number of ghosts G weighted by ω so that it penalises solutions that use more than one ghost as we wanted to keep this number as low as possible. We have experimentally set $\omega = 0.1$ according to the values observed in the other terms of $F(\vec{x})$. The evaluation of the scenarios is done by simulating the system configuration using the ARGoS simulator explained later in Section 4.3.

4.2 Genetic Algorithm (GA)

We have designed a Genetic Algorithm (GA) which uses operators for continuous optimisation, in order find the parameterisation of the formation which keeps the desired polygonal shape as well as the distance to central ghosts. Our decision is justified by the complexity of the solution space as well as the long simulations times required to evaluate the system configurations. The proposed GA is based on an Evolutionary Algorithm (EA) [11]. EA simulates processes present in evolution such as natural selection, gene recombination after reproduction, gene mutation, and the dominance of the fittest individuals over the weaker ones. This is a generational GA where an offspring of λ individuals is obtained from the population μ, so that the auxiliary population contains the same number of individuals (20 in our implementation) as the main population.

We have setup 3,000 evaluations as termination condition, and chosen Binary Tournament [10] as selection operator, Single Point Crossover [7] as recombination operator, Integer Polynomial Mutation [8] as mutation operator, and an elitist replacement. The rest of the parameters of GA are crossover probability $Pc = 0.9$ and mutation probability $Pm = \frac{1}{L}$, all experimentally calculated to obtain a good performance of the algorithm, balancing exploration and exploitation.

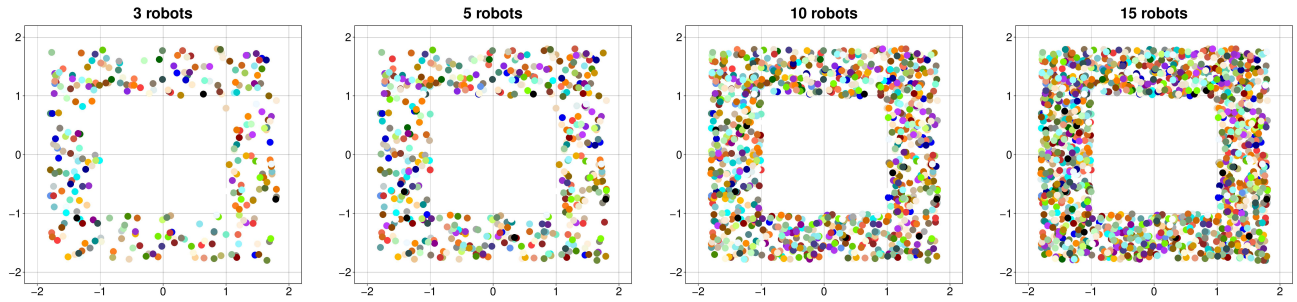


Figure 3: Robots' initial position for the 100 optimisation scenarios per case study.

4.3 Case Studies

The proposed case studies consist of swarms of three, five, ten, and fifteen robots. For each case study there are 100 different scenarios where the robots begin to build the formation from different positions in the area. By doing so, we address a more general problem and the calculated configurations are to be valid for many different situations instead of just a few particular cases. Twenty-eight scenarios per case study were randomly selected for the optimisation process. As described in Equation 3, the resulting fitness value of a given configuration is calculated as the mean of the values of each of the 28 scenarios. Then, once the best configuration is obtained from the optimisation process, it would be tested on the rest (72) scenarios, individually, to study the robustness of the aforementioned configuration, as explained in Section 5.3.

Figure 3 shows the distribution of the initial position of the robots in the predefined 100 optimisation scenarios. Note that the central area is left free as the robots are supposed to arrive from the borders, far from the central point to be surrounded/observed. These scenarios were modelled in ARGoS [20], a multi-physics robot simulator which can simulate large-scale swarms of robots of any kind efficiently. In our study, we simulate the E-Puck2 robots using the Range and Bearing communication model provided by ARGoS. Each robot will only receive a beacon from the others which indicates their relative distance and angle. Although we have worked without any specific length unit as the system can be easily scaled, in ARGoS we have defined an area of 4x4 units and the distances were calculated to fit this scale.

5 EXPERIMENT RESULTS

In this section we describe the experiment setup, the optimisation process performed using the GA, the testing phase where we address the robustness of the achieved solutions, and finally the validation of the results using real robots.

5.1 Experiment Setup

The optimisation process consisted in optimising 28 different scenarios per case study using the proposed GA which was implemented using the jMetalPy package [2]. We have performed 30 independent runs in parallel using computing nodes in the HPC facilities of the University of Luxembourg [24], equipped with Intel Xeon Gold 6132 @ 2.6 GHz and 128 GB of RAM. Since the realistic simulations are costly in terms of execution time, the total optimisation time (120 runs) was equivalent to 242.4 hours (about 10 days).

The controller of the E-Pucks2 robots was programmed so that when there is a change in the moving direction, the robot would stop their translation, rotate until it reaches the new orientation to begin to advance again. The same behaviour was implemented on the actual E-Pucks2 for the final validation of the results.

5.2 Optimisation Results

Table 2 shows the results of the optimisation process. Fitness values are reported from each set of 30 runs, where it can be seen that for three and five robots the GA has found perfect triangles and pentagons respectively (fitness values equal to 0), while for ten and fifteen robots there are still some minimal imperfections in the final shapes (note that fitness values were also penalised due to the use of multiple ghosts in these two cases, so the error is actually around one hundredth). As mentioned, the GA has converged to solutions using more than one ghost for ten and fifteen robots and has succeeded where the geometric approach has failed. The observed precision of GA also denotes that it was not falling into local minima during the optimisation process. The next step was to address the robustness of the achieved solutions by testing them on 72 unseen scenarios per case study.

Table 2: Results of the optimisation process using the proposed GA using 28 scenarios ($D_{centre} = 1.00$).

Robots	D_{robot} (Best)	N_{ghosts} (Best)	Fitness Min.	Fitness Avg.	Fitness Max.
3	1.73	1	0.0000	0.0000	0.0000
5	1.63	1	0.0000	0.0013	0.0200
10	1.57	2	0.1100	0.1101	0.1104
15	1.57	4	0.3107	0.3114	0.3218

5.3 Robustness Analysis

In this section we test the best configuration for each case study on a number of unseen scenarios to know how robust the solutions provided by the GA are. Additionally, we have tested on the same scenarios the geometric approach and another optimised solution

Table 3: Results after testing the different approaches on 72 unseen scenarios. The best results are in bold.

Robots	Approach	N_{ghosts}	$D_{robot}\star$			$D_{centre}\star$			Wilcoxon
			Min.	Avg.	Max.	Min.	Avg.	Max.	p-value
3	Geometric	1	1.719	1.729	1.740	**0.992**	**0.999**	**1.005**	identical
	0-Ghosts	0	1.454	2.727	2.830	0.474	1.691	2.708	2.70×10^{-034}
	Optimised	1	1.719	1.729	1.740	**0.992**	**0.999**	**1.005**	—
5	Geometric	1	0.891	0.899	0.912	0.761	0.767	0.774	9.44×10^{-061}
	0-Ghosts	0	0.807	0.856	1.152	0.089	0.925	2.164	1.26×10^{-005}
	Optimised	1	1.169	1.176	1.187	**0.995**	**1.003**	**1.009**	—
10	Geometric	1	0.244	0.264	0.354	0.618	**0.997**	1.320	9.95×10^{-001}
	0-Ghosts	0	0.414	0.426	0.617	0.032	0.649	1.802	3.88×10^{-101}
	Optimised	2	0.597	0.606	0.631	**0.982**	0.992	**1.001**	—
15	Geometric	1	0.089	0.130	0.367	0.233	0.837	1.850	1.78×10^{-034}
	0-Ghosts	0	0.354	0.381	0.504	0.014	0.690	1.306	3.19×10^{-159}
	Optimised	4	0.403	0.409	0.423	**0.982**	**0.996**	**1.009**	—

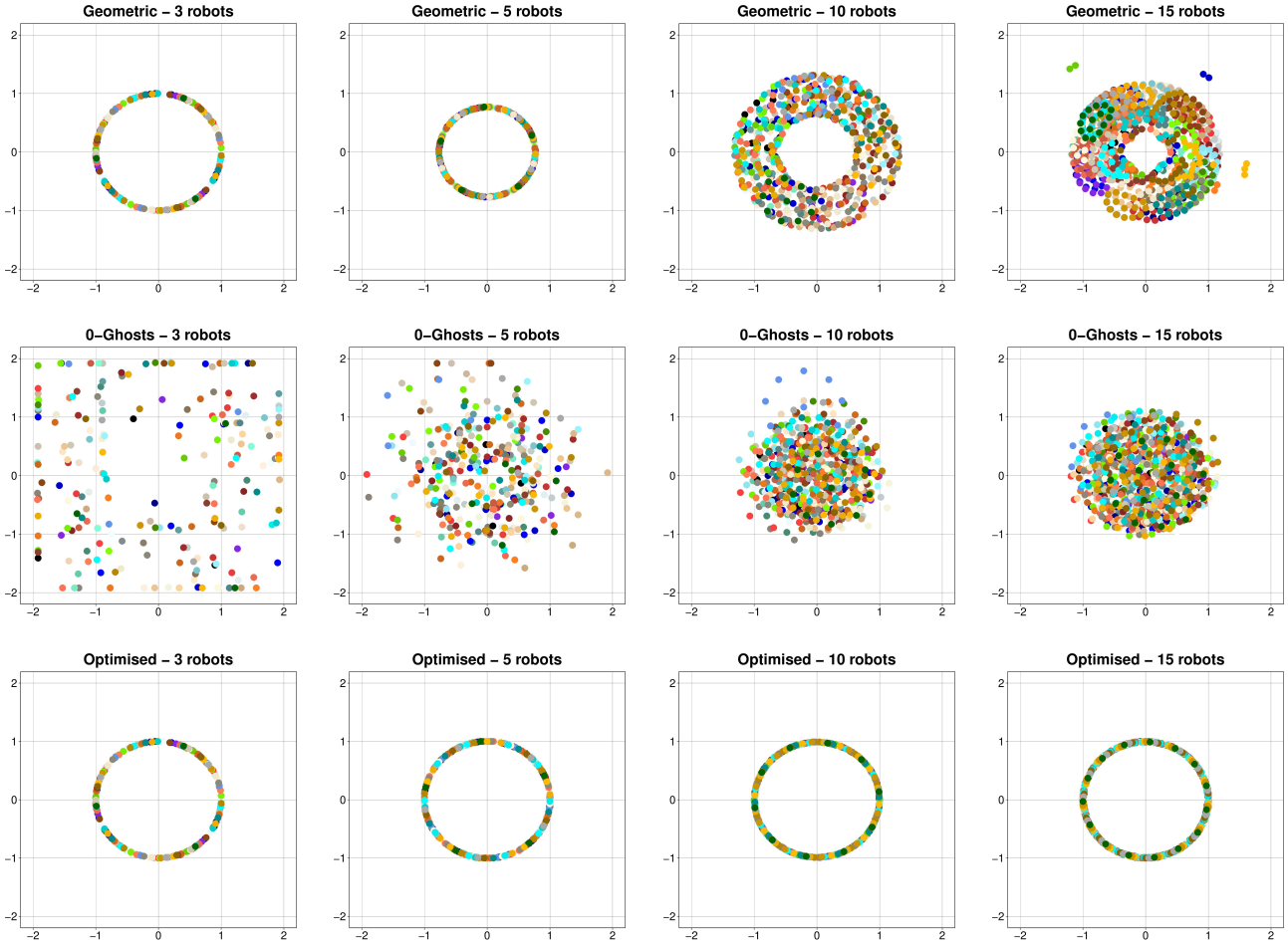


Figure 4: Final positions of the robots (72 scenarios) when using the studied approaches: geometric, 0-ghosts, and optimised.

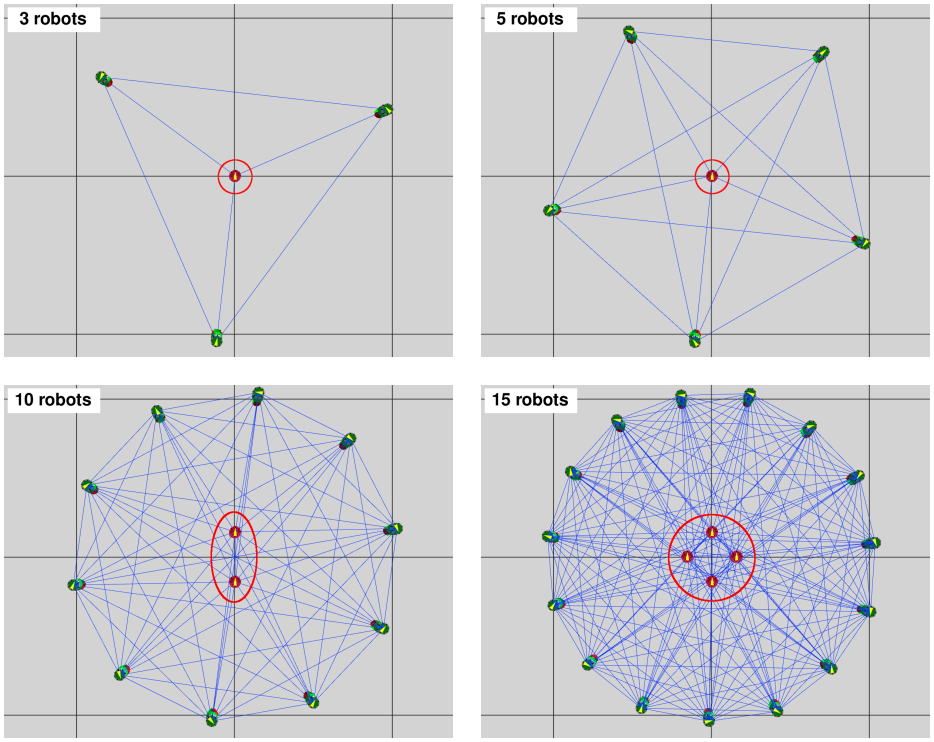


Figure 5: Final formations achieved using the optimised configurations for the robots. All the circles are centred in the ghosts and have a radius equal to 1.0, the desired D_{centre}. Only one testing scenario is shown for clarity.

called 0-Ghosts. By doing so, we want to perform a sanity check and to confirm that the use of central ghosts are essential to achieve the final desired shapes.

Table 3 shows the results obtained after testing the proposed approaches on 72 unseen scenarios per case study. It can be seen that for three robots, the optimised results and the geometric calculation have obtained the same values as the relation of distances between robots and the desired distance to the centre are geometrically compatible. In these cases the measured distance $D_{centre}\star$ is practically the desired value, i.e. 1.0, while the measured distance between robots are the same values for both approaches, i.e. $D_{robot}\star \simeq 1.7$.

Some differences appeared when using five robots. The *Geometric* approach, despite of achieving the desired shape, ended up with a distance to the centre notably lower, i.e. $D_{centre}\star \simeq 0.77$. However, the optimised approach did achieve $D_{centre}\star \simeq 1.0$ with a minimal deviation across the 72 scenarios. In the scenarios having ten and fifteen robots, the optimised approach obtained the best results ($D_{centre}\star \simeq 1.0$) while the *Geometric* approach was unable to obtain such a precise results, presenting a high variability for the 72 scenarios. The *0-Ghosts* approach fails to form a valid shape in most of cases and the measured values show the highest dispersion among the three approaches. All these results have been statistically tested using the Wilcoxon p-value to confirm the existing high grade of significance (p-values $\ll 0.01$) except for the *Geometric* approach for 10 robots, where the average value of ($D_{centre}\star$) is similar to the *Optimised* approach. However, *Geometric*'s minimum and maximum values evidence the lack of precision of this approach.

Finally, Figure 4 shows the final distribution of the robots for each approach and case study. All the testing scenarios were included so that there are 216, 360, 720, and 1080 robots depicted for each case study, respectively. It can be seen how for three robots the *Geometric* approach gets the desired result (all the robots are in the circle's perimeter), but for five robots the radius is clearly lower than 1.0. There is a number of anomalies in the formed shapes for ten and fifteen robots when using *Geometric*, as it was mentioned when the numeric results were analysed. The final positions for the *0-Ghosts* approach were completely chaotic as expected since there is no central point (ghost) to organise the robots around it. Finally, the optimised approach shows four circles of radius 1.00 centred in the central point (0,0) where the ghost(s) were located.

Figure 5 shows the final formations achieved by the robots for the four case studies (just one testing scenario for improving clarity) when using the optimised configuration calculated by the GA. All the robots are keeping the desired distance to the centre as well as the almost perfect polygonal shape. It can be seen how the use of multiple central ghosts plus the optimised distance between robots have overcome the difficulties observed when using the geometric approach, c.f. Figure 2.

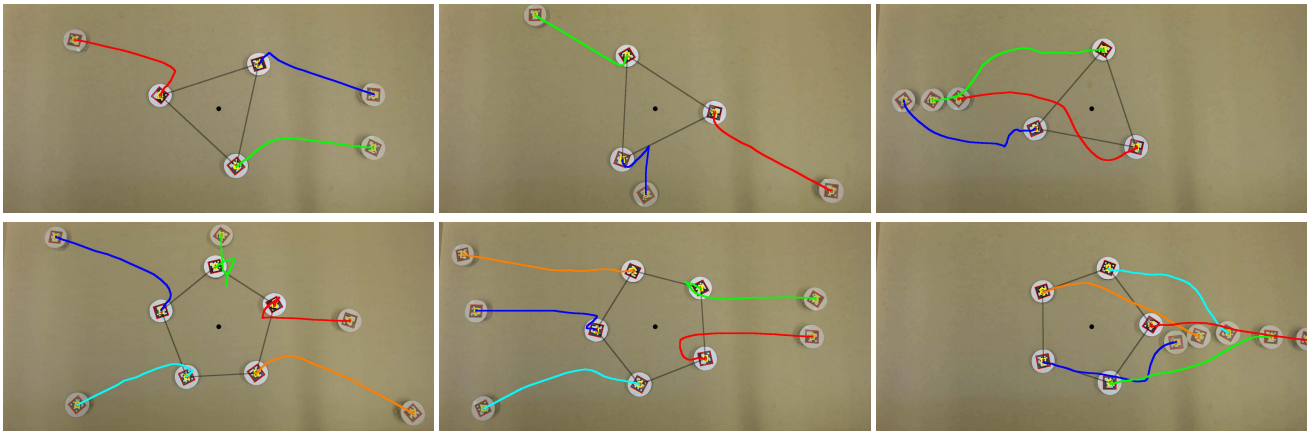


Figure 6: Initial and final positions plus the detailed trajectories followed by the E-Puck2 robots in six scenarios. Video available at https://adars.uni.lu/

5.4 Validation of Results

In order to validate the simulation results using a real world application, we have developed a testing environment using E-Puck2 robots and a video camera (Logitech C925e) to detect the robots' positions and simulate the communication layer, providing range and bearing information.

The E-Puck2 robots, developed by GCtronic [9], are small devices featuring a pair of wheels with encoders, proximity and time of fight (frontal distance) sensors, accelerometers, 3D gyro, magnetometers, VGA Camera, etc. The connectivity is done through USB, BLE, WiFi or Bluetooth. Each robot is identified by a different Aruco [21] code which is used to detect its position and orientation using opencv [19]. These permit to calculate the range and bearing data which are then provided to each robot's controller. Then, the robot's wheels are controlled via WiFi link, being able to rotate a given number of degrees or move forward at constant speed. As in the simulations, the robots do not know their own absolute position nor the others' coordinates, just the relative range and bearing from the other robots.

Figure 6 shows the initial position and the trajectories followed by the robots in six different experiments using three and five E-Pucks. In every case, there is a simulated ghost to define the central point of the screen from where the robots keep the fixed distance to the centre when forming the shape. We have linearly scaled the distances obtained during the optimisation to adapt our solutions to the real world experiment. Given the desired $D_{centre} = 25$, we have calculated $D_{robot} = \lceil 25 \times 1.73 \rceil = 44$ and $D_{robot} = \lceil 25 \times 1.63 \rceil = 41$ for three and five robots, respectively. During the experiments we have observed that the camera has temporary lost focus in some occasions so that the robots' positions were not updated constantly. This has given us some insight into the robustness of our proposal with respect to temporary communication loss, since the global performance of the system was not affected and the final formation was always achieved.

6 CONCLUSIONS AND FUTURE WORK

In this article we have proposed a novel system to control a swarm of autonomous robots developing formation tasks, e.g. surrounding a rogue drone or observing asteroids. Our proposed algorithm achieves stable robot formations around a predefined central point and distance. We have proposed four case studies including three, five, ten, and fifteen robots and also designed a genetic algorithm to calculate the optimal parameters of the system, by evaluating 28 training scenarios per case study, in parallel. We have tested our results on 288 unseen scenarios to simulate real world situations in which the arriving positions of the swarm members are unknown. By doing so, we have evaluated the robustness and reliability of the configurations, where our proposal has achieved the desired formation in 100% of scenarios. Finally, we have conducted experiments with real robots to validate our simulation results by scaling the obtained configuration values to real world magnitudes.

As a matter of future work we would like to address obstacles and communication loss to challenge our formation algorithm and evaluate the time elapsed by the robots to be at their final positions as another metric to be taken into account. Since using several ghost could be difficult to implement in a final application we are working on replacing them by a parameterised repelling force to preserve the formation stability. We wish to test formations following a moving centre and dynamically add/remove robots to the swarm. After that, once our formation system is mature, we plan to extend our proposal to 3D space adding a third component to the forces involved in the model and test it using simulations and actual quad rotor drones.

ACKNOWLEDGEMENTS

This work is supported by the Luxembourg National Research Fund (FNR) – ADARS Project, ref. C20/IS/14762457. The experiments presented in this paper were carried out using the the SwarmLab facility of the FSTM/DCS and the HPC facilities of the University of Luxembourg [24] – see https://hpc.uni.lu.

REFERENCES

[1] Javier Alonso-Mora, Eduardo Montijano, Tobias Nägeli, Otmar Hilliges, Mac Schwager, and Daniela Rus. 2019. Distributed multi-robot formation control in dynamic environments. *Autonomous Robots* 43, 5 (2019), 1079–1100. https://doi.org/10.1007/s10514-018-9783-9

[2] Antonio Benítez-Hidalgo, Antonio J. Nebro, José García-Nieto, Izaskun Oregi, and Javier Del Ser. 2019. jMetalPy: A Python framework for multi-objective optimization with metaheuristics. *Swarm and Evolutionary Computation* (2019), 100598. https://doi.org/10.1016/j.swevo.2019.100598

[3] Matthias R. Brust, Grégoire Danoy, Daniel H. Stolfi, and Pascal Bouvry. 2021. Swarm-based counter UAV defense system. *Discover Internet of Things* 1, 1 (feb 2021). https://doi.org/10.1007/s43926-021-00002-x

[4] Gustavo A. Cardona and Juan M. Calderon. 2019. Robot swarm navigation and victim detection using rendezvous consensus in search and rescue operations. *Applied Sciences* 9, 8 (2019). https://doi.org/10.3390/app9081702

[5] Soon-Jo Chung, Umair Ahsun, and Jean-Jacques E. Slotine. 2009. Application of synchronization to formation flying spacecraft: Lagrangian approach. *Journal of Guidance, Control, and Dynamics* 32, 2 (mar 2009), 512–526. https://doi.org/10.2514/1.37261

[6] Saar Cohen and Noa Agmon. 2021. Recent advances in formations of multiple robots. *Current Robotics Reports* 2, 2 (2021), 159–175. https://doi.org/10.1007/s43154-021-00049-2

[7] Kenneth Alan De Jong. 1975. *An analysis of the behavior of a class of genetic adaptive systems.* Ph.D. Dissertation.

[8] Kalyanmoy Deb. 2001. *Multi-objective optimization using evolutionary algorithms.* John Wiley & Sons, Inc., USA.

[9] GCtronic. 2022. GCtronic – Electronics and mechatronics. (Jan. 2022). https://www.gctronic.com/

[10] David E Goldberg and Kalyanmoy Deb. 1991. A comparative analysis of selection schemes used in genetic algorithms. *Foundations of Genetic Algorithms* 1 (1991), 69–93. https://doi.org/10.1016/B978-0-08-050684-5.50008-2

[11] John Henry Holland. 1992. *Adaptation in natural and artificial systems.* The MIT Press. 228 pages. https://doi.org/10.7551/mitpress/1090.001.0001

[12] Bayadir A. Issa and Abdulmuttalib T. Rashid. 2019. A survey of multi-mobile robot formation control. *International Journal of Computer Applications* 181, 48 (Apr 2019), 12–16. https://doi.org/10.5120/ijca2019918651

[13] Chao Jiang, Zhuo Chen, and Yi Guo. 2019. Learning decentralized control policies for multi-robot formation. In *2019 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM).* 758–765. https://doi.org/10.1109/AIM.2019.8868898

[14] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.

[15] Yuanchang Liu and Richard Bucknall. 2018. A survey of formation control and motion planning of multiple unmanned vehicles. *Robotica* 36, 7 (2018), 1019–1047. https://doi.org/10.1017/S0263574718000218

[16] Satoshi Makita and Weiwei Wan. 2017. A survey of robotic caging and its applications. *Advanced Robotics* 31, 19-20 (2017), 1071–1085. https://doi.org/10.1080/01691864.2017.1371075

[17] Nicholas Metropolis and Stanislaw Ulam. 1949. The Monte Carlo method. *J. Amer. Statist. Assoc.* 44, 247 (Sept. 1949), 335–341. https://doi.org/10.1080/01621459.1949.10483310 Publisher: Taylor & Francis.

[18] Sérgio Monteiro and Estela Bicho. 2010. Attractor dynamics approach to formation control: theory and application. *Autonomous Robots* 29, 3-4 (jul 2010), 331–355. https://doi.org/10.1007/s10514-010-9198-8

[19] OpenCV Team. 2022. OpenCV. (Jan. 2022). https://opencv.org/

[20] Carlo Pinciroli, Vito Trianni, Rehan O'Grady, Giovanni Pini, Arne Brutschy, Manuele Brambilla, Nithin Mathews, Eliseo Ferrante, Gianni Di Caro, Frederick Ducatelle, Mauro Birattari, Luca Maria Gambardella, and Marco Dorigo. 2012. ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm Intelligence* 6, 4 (dec 2012), 271–295. https://doi.org/10.1007/s11721-012-0072-5

[21] Francisco J. Romero-Ramirez, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer. 2018. Speeded up detection of squared fiducial markers. *Image and Vision Computing* 76 (aug 2018), 38–47. https://doi.org/10.1016/j.imavis.2018.05.004

[22] Sajad Saeedi, Michael Trentini, Mae Seto, and Howard Li. 2015. Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics* 33, 1 (jul 2015), 3–46. https://doi.org/10.1002/rob.21620

[23] Jun Tang, Gang Liu, and Qingtao Pan. 2021. A review on representative swarm intelligence algorithms for solving optimization problems: applications and trends. *IEEE/CAA Journal of Automatica Sinica* 8, 10 (2021), 1627–1643. https://doi.org/10.1109/JAS.2021.1004129

[24] Sebastien Varrette, Pascal Bouvry, Hyacinthe Cartiaux, and Fotis Georgatos. 2014. Management of an academic HPC cluster: The UL experience. In *2014 International Conference on High Performance Computing & Simulation (HPCS).* IEEE, Bologna, Italy, 959–967. https://doi.org/10.1109/HPCSim.2014.6903792

[25] Shuo Wan, Jiaxun Lu, and Pingyi Fan. 2017. Semi-centralized control for multi robot formation. In *2017 2nd International Conference on Robotics and Automation Engineering (ICRAE).* 31–36. https://doi.org/10.1109/ICRAE.2017.8291348