Adaptive Self-assembly in Swarm robotics through Environmental Bias

Jean-Marc Montanier
CRAB Lab,
Department of Computer and Information Science,
NTNU, Trondheim, Norway
jean-marc.montanier@idi.ntnu.no

Pauline C. Haddow
CRAB Lab,
Department of Computer and Information Science,
NTNU, Trondheim, Norway
pauline@idi.ntnu.no

***Abstract*—A swarm of robots may face challenges in unknown environments where self-assembly is a necessity e.g. crossing difficult areas. When exploring such environments, the self-assembly process has to be triggered only where needed and only for those robots required, leaving other robots to continue exploration. Further, self-assembled robots should dis-assemble when assembled structures are no longer required. Strategies have thus to be learned to trigger self-assembly and dis-assembly so as to meet the needs of the environment. Research has focused on the learning of strategies where all robots of the swarm had to adopt one common strategy: either self-assembly or dis-assembly. The work herein studies how strategies using both self-assembly and dis-assembly can be learned within the same swarm. Further, the effect of the different environments on this challenge is presented.**

I. Introduction

A swarm of robots can face environmental challenges such as crossing a difficult obstacle or moving a heavy object. In order to address such challenges, it has been proposed to endow the robots with the capacity to self-assemble in larger structures [1]. Triggering such self-assembly through environmental cues is termed *functional self-assembly* [2]. Despite the benefits of self-assembly, whilst some or all robots are assembled, less robots are available to explore the area, an issue for time-critical exploration scenarios in the real world. Thus so as to maximise efficient exploration, self-assembly should only occur where and when needed and only with the number of robots required so as to meet the needs of the environment, leaving a maximum number of individual robots available for exploration. Further, self-assembly and dis-assembly may be required in different areas of the environment at the same time and such processes should be conserved until no longer required.

When a swarm of robots is deployed in the rubble of collapsed buildings, self-assembled robotic structures are needed to bridge over obstacles and should be dis-assembled when the obstacle is crossed. In such a time critical scenario it is clear that those robots that are not needed for the structure required, should be free to explore. Similarly, where a swarm of robots that are deployed to assist a person, they can meet a heavy object to be moved, requiring sufficient robots to self-assemble into a strong enough structure. Again those not required should be conserved as dis-assembled and thus free to explore for further tasks. In both these applications, the exact environment is not specified *a priori* and may be composed of many sub-environments each with their own needs with regards to the self-assembly and dis-assembly processes. As a consequence, the robots of the swarm will have to learn autonomously different strategies adapted to the environment at hand. In the work herein the learning of strategies capable of self-assembly and dis-assembly in unknown environments is studied in depth.

In section II works related to the learning of self-assembly are presented. The evolutionary algorithm used is described in section III. The experimental set-up is presented in section IV, followed by the results obtained. Finally the conclusions are presented in section VI.

II. Related Work

The Evolutionary Robotic (ER) domain studies the automated design of robot strategies by relying on the principles of artificial evolution [3]. Once the design process is completed, the best strategies can be used on real robots to perform the task at hand. Research addressing functional self-assembly has focused on the design of explicit fitness functions [4], [5]. These fitness functions measure the robot's ability to learn a strategy for solving a specific well defined task in a known environment. If the task or environment changes, a new strategy would need to be evolved, requiring a new fitness function.

To face such a challenge, fitness functions can, instead, be expressed in terms of rewards. In [2], [6], different sub-environments of the environment give rewards of different amounts, the highest reward given where self-assembly is required. To observe evolution of self-assembly strategies, the fitness function thus maximises the rewards achieved, driving all the robots to the sub-environment where self-assembly is rewarded. Such an approach does not provide for dis-assembled robots in the swarm and further requires the designer to know where self-assembly is needed in the environment. Autonomous adaptation to the environment is the focus of Embodied Evolutionary Robotics [7] where fitness functions use information locally available to the robots and the evolutionary algorithm runs while the robots are deployed in the environment. In [8], a reward-based evolution of self-assembled structures, the results highlighted the necessity to give higher rewards to self-assembled robots.

In [9] a fitness function rewards the robots travelling the furthest away in an environment where some of the sub-environments can only be crossed by self-assembled robots and the others by dis-assembled robots. This approach results in the presence of all the robots in the last sub-environment where they will all use the same strategy, thus hindering the exploration abilities of the swarm.

In the work herein, investigation of the dispersion of the swarm and the achievement of self-assembly and dis-assembly is conducted so as to determine whether the conditions for self-assembly and dis-assembly can in fact depend on local environmental constraints and thus determine whether self-assembly rewards and/or fitness controlled movements are necessary.

III. Evolutionary Algorithm

The mEDEA algorithm [10] addresses the evolution of robot strategies without the need for an explicit fitness function. The evolutionary process evolves and adapts the strategies of the robots to the constraints of the environment, in situ.

Within mEDEA each robot contains an *active* genome and a *reservoir of received genomes*. An active genome encodes the parameters of the current controller i.e. each robot's strategy depends on the robot's current active genome. At first the active genome is generated randomly, and the reservoir is empty. At any time step, each robot *broadcasts* in a limited range a mutated copy (gaussian mutation) of its active genome and stores genomes received from neighbours. Only one copy of each received genome is stored in the reservoir. At the end of a generation i.e. a pre-defined number of time steps, each robot "forgets" its active genome and *randomly* picks one genome from its reservoir of stored genomes (if not empty). If the reservoir is empty the robot becomes inactive. An inactive robot remains stationary and stores broadcasted genomes during one generation after which it attempts to select a new active genome. When a new genome is successfully selected, the reservoir of the robot is emptied. This algorithm is duplicated within each robot in the population.

The working of this algorithm is illustrated in figure 1. In (1), a generation starts, and the reservoir of genomes of each robot is empty. Two robots are controlled by their respective active genomes (in bold): G1 and G2. The robot in the top right corner is inactive. In (2) and (3) the robots with an active genome move in the environment and the inactive robot remains stationary. In (2) the active robots are close and therefore store each other's mutated genomes in their respective reservoir of received genomes (in grey). In (3) the robot controlled by G2 is close to the inactive robot, which stores a mutation of G2 in its reservoir. In (4), the current generation ends: the genome G2 has spread more and thus has higher probability of being selected. One robot is using a local modification of the genome G1, termed G1', two robots are using local modifications of G2 termed G2'.

For a genome to survive, it needs to spread to other robots so as to be present in their reservoir. Additionally, a genome which has spread more than the others has a higher chance to be selected. Notice that the selection of the genomes does not depend on a fitness function but on the interaction between the robots. These interactions further depend on the environment.

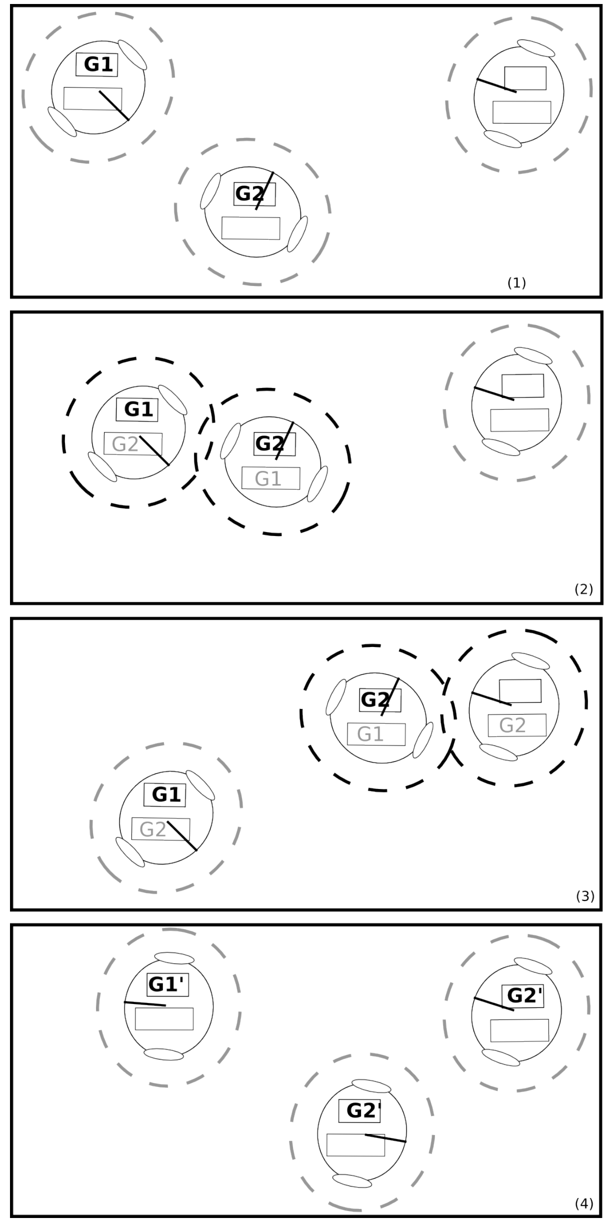

Fig. 1: The mEDEA algorithm: a simplified illustration.

IV. Experimental Setup

The work herein investigates the conditions necessary for the evolution of strategies enabling both self-assembly and dis-assembly within a given environment. The first hypothesis, inspired by the work of [8] providing higher rewards for self-assembled robots, investigates how such increased rewards can promote the evolution of strategies relying on both self-assembly and dis-assembly and without the introduction of sub-environments. The second hypothesis proposes that the introduction of sub-environments alone i.e. without rewards, can promote both self-assembly and dis-assembly strategies within an environment. All experiments are performed with 100 robots using the mEDEA algorithm and Roborobo — a fast open-source multi-robot simulator [11]. The parameter settings are given in table I and the full implementation of the experiments is available [12].

Parameter	*Value*
arena width and length	$1024 * 530$ pixels
population size	100 robots
lifetime (i.e. generation duration)	400 steps
proximity sensor range	64 pixels
broadcast signal	32 pixels
self-assembly distance	20 pixels
robot rotational velocity	0.52 rad/time step
robot translational velocity	2 pixels/time step
genome length	82 real values
# food items	800
energy item diameter	10 pixels
energy item regrow delay	25
energy per energy point	50 energy units
energy per biased energy point	500 energy units
robot energy consumption	1 energy unit per step
robot maximum energy level	800 energy units
robot initial energy level	400 energy units

TABLE I: Parameter Settings

A. *Energy Implementation*

Robots are endowed with a battery of limited capacity and are loosing one unit of energy per time step. The battery is pre-loaded with an initial amount of energy. If a robot runs out of energy it will become inactive. This comes in addition to the inactivity triggered when no genomes are present in the reservoir at the end of a generation (see section III). An inactive robot stops moving and does not spread its genome.

In order to remain active a robot has to harvest energy from energy points present in the environment. When an energy point is harvested it disappears for a fixed delay and its energy is added to the battery of the robot. If the energy points contain more energy than necessary to refill the battery, the excess is lost. In order to assume simple and physically plausible assumptions, self-assembled robots do not exchange energy between each other. Therefore, when a robot harvests an energy point, all the energy goes to its battery.

When inactive, a robot receives the genomes distributed by neighbouring robots during a given generation. If a genome has been received, the robot becomes active and its battery is loaded with an initial amount of energy.

In the work herein, the genomes have to maintain the robots active in order to be spread in the swarm. In other words, they have to adopt a strategy which leads the robots to harvest energy. The presence of energy points in the environment depends both on the strategies used by each robot (under evolution) and on the specificities of the energy points (size, number, energy in each). The specificities of the energy points are set so as to produce a relatively low constraints on energy access (see table I).

B. *Robots*

The simulated robots used in this work are based on the ChIRP robot [13]. Each robot has a round body of 2 pixels in diameter and can move at a maximum velocity of 2 $pixels/timesteps$. 2 actuators are connected to the wheels. By slowing down either actuator the robot will turn in the corresponding direction at an angle proportional to the difference in actuator speeds. There are eight evenly distributed proximity sensors, as depicted in figure 2. Each proximity sensor has a range of 64 pixels, thus able to measure the distance to obstacles within this range. In addition to the sensors provided by the platform, 4 further sensors are simulated: 2 sensors return the direction and distance to the closest energy point; one sensor returns a binary value depending on the position of the robot (on top of an energy point or not) and one sensor within the robot reads the amount of energy left in the battery. Each robot is, further, endowed with a mechanism to self-assemble with other robots. This assembly mechanism is based on a such a prototyped extension for the ChIRP and has been shown to be physically realistic.

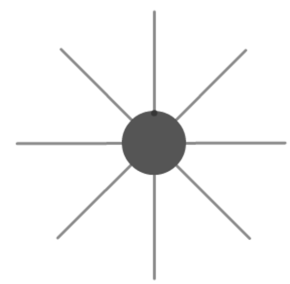

Fig. 2: Placement of distance sensors of the robots

Intuitively if two robots are to assemble, both should have an intent to assemble. However, such a strict requirement reduces the chance of assembly, requiring a robot wishing to assemble to not only meet another robot but one with intent to assemble and within the time that the robot's assembly intent is still active. As such, one can expect little or no self-assembly in the earlier phase of any experiment. Strategies for dis-assembly will then be optimised and those for self-assembly will be disregarded. To counteract this effect, the physical prototype for assembly was designed to enable a robot, with the intent to self-assemble (assembly mechanism activated), to assemble with other robots that it by meets by chance i.e. no requirement for self-assembly intent by the other robot. If the state of the assembly mechanism of an assembled robot changes from activated to deactivated the two robots will dis-assemble. Groups of self-assembled robots can further self-assemble and dis-assemble with each others and with dis-assembled robots. For simplicity, it is assumed that the movement of a group of self-assembled robots corresponds to the linear sum of the individual movements of its members.

C. *Robot's Controller*

The controller of a robot is composed of a local copy of the mEDEA algorithm and a feed-forward neural network. The current active genome sets the weights of the feed-forward neural network. The neural network reads the values returned by the sensors and sets the activation value of the actuators and the assembly mechanism. Therefore, by optimizing the genomes, the evolutionary algorithm modifies the strategies of the robots. When the robot is inactive, no active genomes are present and the robot stands still.

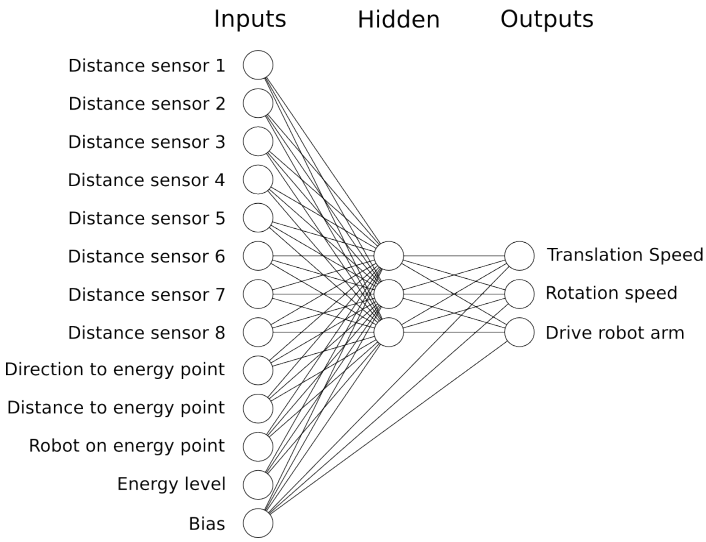

Fig. 3: Feedforward network controlling the robot

The feed-forward neural network is illustrated in figure 3. It is composed of 13 inputs, 3 hidden neurons and 3 outputs. The inputs are the values of 12 sensors present on the robot and 1 neuron is biased to 1. The 12 sensors are: 8 distance sensors, 1 sensor returning the direction to the closest energy point, 1 sensor returning the distance to the closest energy point, 1 binary sensor indicating the presence of the robot on an energy point and 1 sensor returning the level of energy in the robot. The inputs are all scaled between 0 and 1. The two first outputs are used to control the translation speed (TS) and rotation speed (RS) of the robot. The translational speed is scaled between 0 and TS_{Max}. The rotational speed is scaled between 0 and RS_{Max}. The last output (between 0 and 1) controls the activation of the self-assembly mechanism. If the output is less than 0.5 the self-assembly mechanism is activated. If the output is greater or equal to 0.5 the self-assembly is deactivated.

D. Environments

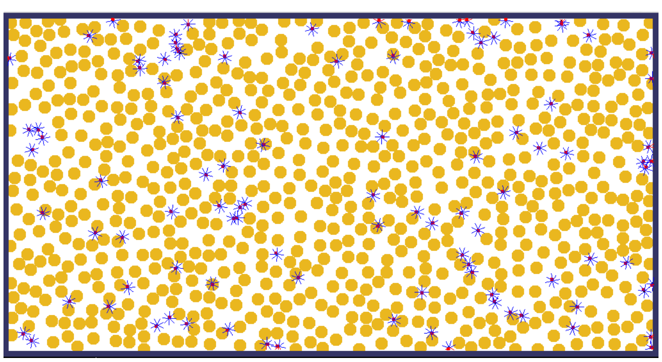

Fig. 4: Screenshot of the experimental setup. The starred crosses are the robots. The disks are the energy points.

The environments simulated herein consist of an arena with 100 robots and 800 energy points – see figure 4. The parameters used to perform the experiments are given in table I. In each environment 200 runs of 400.000 time steps are performed. The properties of the energy points depend on the assumption tested.

1) Baseline Environment: A baseline environment is designed in order to provide a reference point to study the dynamics of the evolutionary algorithm. It may be defined as one where the strategies studied are placed in equally difficult situations i.e. nothing in the environment favours one or the other. The impact of modifications brought to the design of the environment can then be studied with regards to this reference.

The design of the *baseline* environment is inspired by the one proposed in [8] where all robots receive equal rewards independent of their position or whether they are part of a self-assembled structure. For the purposes of the experiments herein the two environmental factors considered are the interaction between robots and the interactions between robots and energy points. The position of robots and energy points are initialized following a uniform random distribution. Further, in order to not introduce bias in the interactions between robots and energy points, the environment features energy points available across all the environment for all robots (self-assembled or dis-assembled).

2) Biased Environments: Biased environments are designed to promote the evolution of self-assembly within the swarm where required whilst enabling dis-assembly of robots elsewhere. As stated, two biases are studied: increasing the reward to self-assembled robots, and creating sub-environments favourable to either self-assembled robots or dis-assembled robots.

These biases are implemented by creating two types of energy points: the ones that can be harvested only by dis-assembled robots, and the ones that can be harvested only by self-assembled robots. The properties and repartitions of these energy points changes with the bias studied.

Energy bias: In this biased environment, the energy points reserved for self-assembled robots contains 10 times more energy than the energy points reserved for dis-assembled robots. This reward is chosen as it is superior to the one proposed in [8] under the linear and logarithmic schemes. Both types of energy points are found in equal proportions, and all energy points have the same size.

Repartition bias: In order to study this bias two environments are created where energy points are found in equal proportions, have the same size and contain the same amount of energy. It is therefore expected to observe half of the robots dis-assembled and half of the robots self-assembled. The energy points are more or less distinctively separated, as illustrated in figure 5. In the *separated* environment each type of energy points are found in their half of the arena. In the *mixed* environment the two types of energy points are uniformly distributed across the arena.

V. Results

A. Baseline Environment

In order to establish a reference on the evolution of self-assembly and dis-assembly, experiments are first run in the baseline environment. The proportion of robots self-assembled (SA_p) has been extracted for each run every 40.000 time

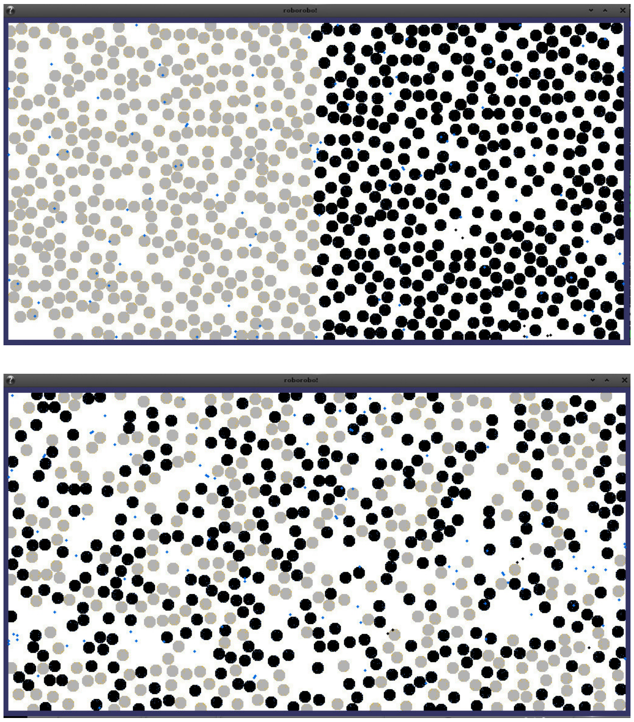

Fig. 5: The grey points are reserved for disassembled robots and the black points are reserved for self-assembled robots. Top: Separated environment. Bottom: Mixed environment

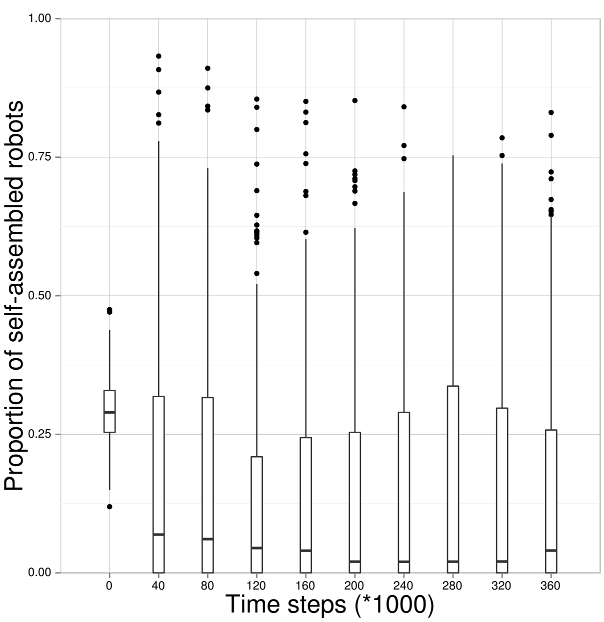

Fig. 6: Proportion of self-assembled robots in the baseline environment.

steps (100 generations of 400 time steps each). Note that the proportion of robots dis-assembled is directly deduced as $1 - SA_p$.

Figure 6 shows the distribution of the results as boxplots. In this graph the lower quartile represents the 25th percentile and the upper quartile represents the 75th percentile. The extremity whiskers extend to 1.5 of the inter-quartile range from their respective quartiles. The data beyond the whisker are considered as outliers and are plotted as points. The center bold mark shows the median of the distribution.

In the first time steps, the median of the distribution is equal to 0.29. The median value is continuously decreasing until stabilizing at a low value (0.04) at time step 200.000. The quartiles are relatively close to the median in the first boxplot, and show large variations in the remaining boxplots. This indicates that the distribution analysed present large variations. The whiskers of the boxplots are further away from the median, and outliers are present in multiple boxplots. These elements confirm the presence of distributions with large variations.

The analysis of the variation of the median shows that in the baseline environment the dis-assembly strategy is preferred over the self-assembly strategy by the evolutionary algorithm. The analysis of the quartiles, whiskers and outliers of the boxplots shows that a large variation is present in the evolution of self-assembly i.e. the runs performed have a large range of outcomes.

This result is coherent with the previous observations made in [8] where dis-assembly is preferred over self-assembly in the absence of rewards for self-assembly strategy. It is now possible to compare the results obtained when an additional reward is given to self-assembled robots with the results obtained in the baseline.

B. Energy Bias

Figure 7 depicts the proportion of self-assembled robots in experiments conducted in the *energy biased* environment. In the first time steps the median of the distribution is equal to 0.36. The median value is decreasing to stabilize at a low level (0.01) from time step 160.000. This low value then remains stable.

This result is similar to the one obtained in the baseline environment. Contrary to expectations, given the work of [2], [9], the *energy biased* environment in fact promotes the evolution of dis-assembly behaviours. To investigate why such a contrary result has appeared, the sub-environments present in the work in [2], [9] are further investigated.

C. Effect of Sub-environments

To study the effect of sub-environments favourable to different strategies, experiments are conducted in the *mixed* and *separated* environments. Recall that in these environments the self-assembled and dis-assembled robots obtain the same amount of energy, but from different types of energy points.

Figure 8 shows the histogram of the proportion of self-assembled robots in the last generations in the *mixed* and *separated* environments. Since the data obtained in one generation are too few to produce a reliable analysis, the data from the last 16.000 time steps (40 generations) of each run are used. In black the results for the *mixed* environment are found, and in grey the results for the *separated* environments are presented.

In the *mixed* environment, most of the runs show a low proportion of self-assembled robots. The distribution show a

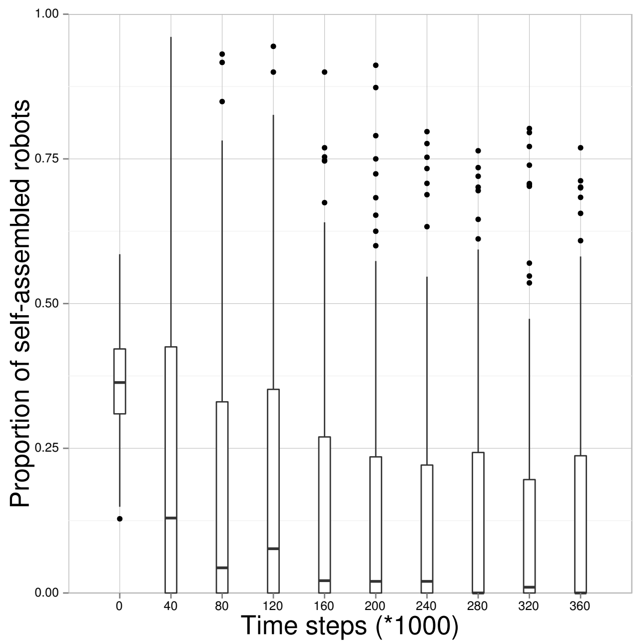

Fig. 7: Proportion of self-assembled robots in the energy biased environment.

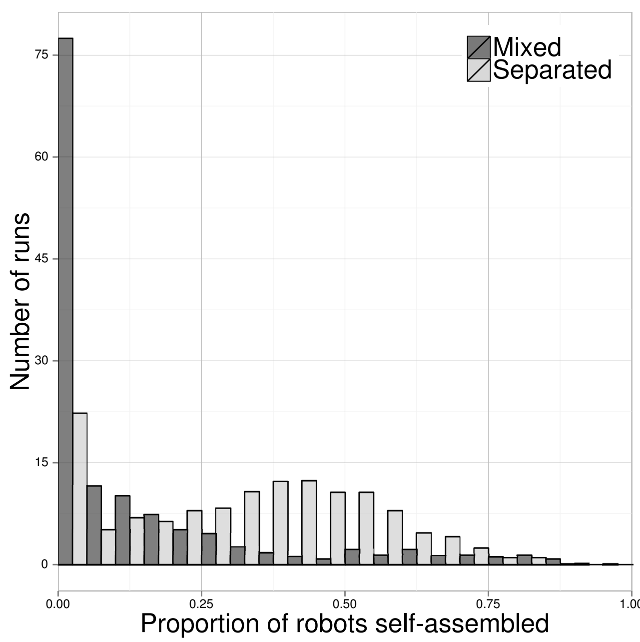

Fig. 8: Proportion of robots self-assembled in the 40 last generations in a mixed environment (black) and a separated environment (grey).

long tail towards higher proportions of self-assembled robots. In the *separated* environment, two local maxima are found: one for a low proportion of self-assembled robots, and one for a proportion of self-assembled robots equal to 0.40. These two distributions are qualitatively and significantly different (wilcoxon test, $p - value < 0.01$).

These results show that, in the *mixed* environment, populations using strategies relying on dis-assembly are more likely to evolve than populations using self-assembly. On the other hand, in the *separated* environments, it is possible to observe the evolution of self-assembly by a part of the population while the other part remain dis-assembled. Therefore, this validates our expectations on the effect of sub-environments in the environment: the presence of sub-environments can promote the evolution of self-assembly and dis-assembly even if no additional rewards are given.

If the swarms were addressing at best the constraints of the environment, only one maxima would be found in the *separated* environment for a proportion of robots self-assembled equal to 0.5. Two avenues are explored in order to understand the presence of a large number of runs relying principally on dis-assembled robots: what is the evolutionary dynamic in the *separated* environment ? How do the strategies used in *mixed* and *separated* environments differ ?

D. Evolution of Self-assembly

Figure 9 shows the proportion of self-assembled robots in the 200 runs performed in the *separated* environment. Looking at the median value, the proportion of self-assembled robots is equal to 0.23 in the first time steps. This value then increase, and fluctuates but never goes under 0.25.

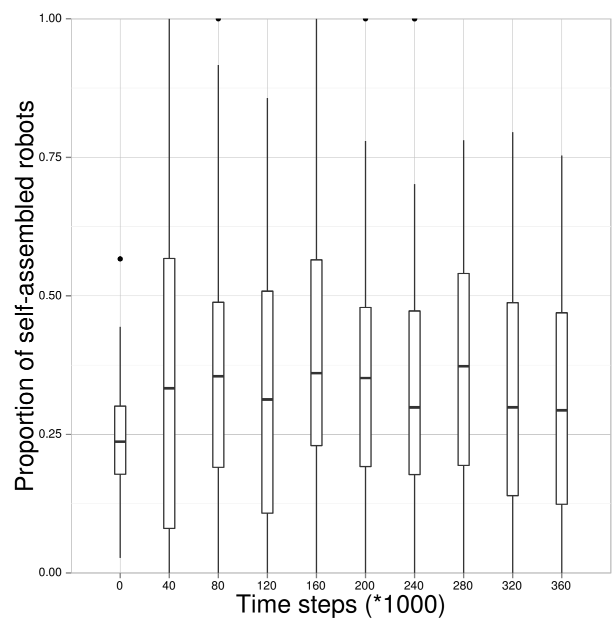

Fig. 9: Proportion of self-assembled robots in a separated environment.

This result shows that the strategies using self-assembly are present in the first time steps, as in the *baseline* environment. However, the populations evolved in the *separated* environment show an increasing rate of self-assembled robots. This shows that in the *separated* environment the evolutionary dynamic tends to address the environmental constraints.

However the medians of the boxplots never reach values higher than 0.40, rather than the 0.5 that may be expected.

Thus interactions between the robots and the environment are still biasing dis-assembly strategies and require further investigation.

E. Strategies with Self-assembly

The strategies evolved in the separated environment are compared to the ones evolved in the mixed environment on two aspects: the use of the assembly mechanism and the movement of the robots. This aims to show the effect of the separation of the environment on the strategies evolved.

The use of the assembly mechanism is analysed first. Recall that this mechanism is activated or deactivated by the controller of a robot. When robots are within self-assembly distance of one another, they automatically self-assemble if at least one robot has activated its assembly mechanism. The connection between robots holds as long as one of them has the assembly mechanism activated.

If the robots of a swarm seldom activate their assembly mechanisms, there will be a low probability to observe self-assembled structures. On the other hand, if the robots of a swarm often activate their assembly mechanism, there will be a high probability to observe self-assembled structures. Nevertheless, this probability will not be equal to 1 since two robots have to come within self-assembly distance in order to self-assemble.

Figure 10 shows the histogram of the activation duration of the assembly mechanisms (value normalised by the lifetime of the robots). The durations obtained during the last 40 generations of the runs performed in the mixed and separated environments are displayed in black and grey, respectively.

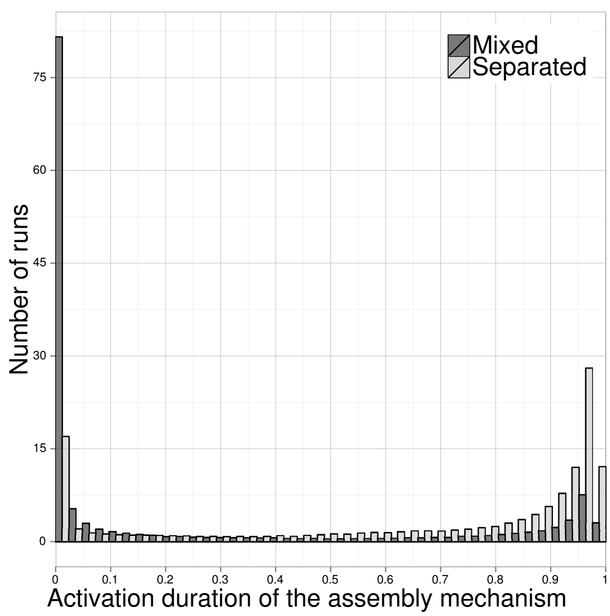

Fig. 10: Portion of time during which robots have their assembly mechanism activated.

The histogram in a *mixed* environment has two local maxima: one for short activation durations and one for long activation durations. The first maxima is higher than the second. The histogram in a *separated* environment shows two similar local maxima, but the second local maxima is higher than the first.

These results show that two different usages of the self-assembly mechanism are evolved: seldom activated and often activated. The first usage is preferred in the *mixed* environment (low number of self-assembled robots) while the second is preferred in the *separated* environment (higher number of self-assembled robots). This shows that the activation duration of the self-assembly mechanism is effectively used so as to regulate the number of self-assembled and dis-assembled robots depending on the environment.

In the separated environment, despite long activations of the self-assembly mechanisms, more robots are dis-assembled than self-assembled (see figure 9). A possible interpretation is that since robots can not locate each other, they have to rely on chance to come within self-assembly distance of other robots. Therefore, a robot may have to move a long time before being able to self-assemble.

The second aspect investigated in the strategies of the robots is their movements. In the *separated* environment (depicted in figure 5 top), if the movements of the robots exploit the specificity of the environment, a majority of the self-assembled robots will be found in the right side where energy points reserved for self-assembled robots are found.

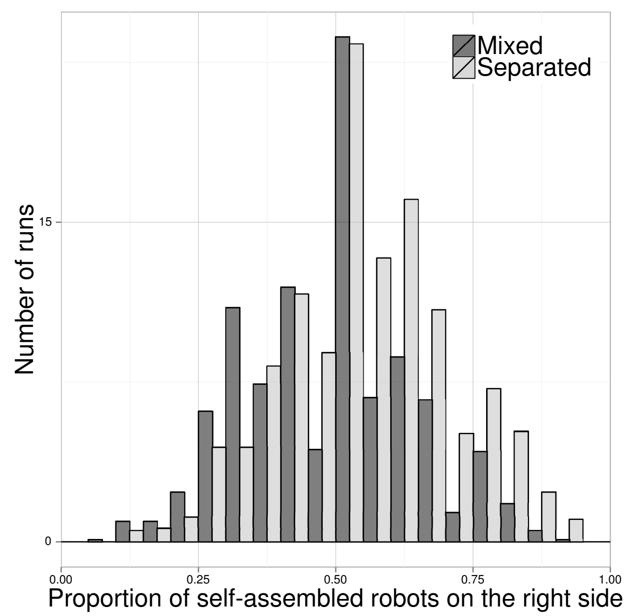

Fig. 11: Proportion of self-assembled robots on the last 40 generation in the separated environment and the mixed environment.

The proportion of self-assembled robots on the right side is shown in figure 11. The data from the last 40 generations are used. The x-axis shows the proportion of self-assembled robots on the right side of the environment i.e. the proportion of self-assembled robots on the side of the environment favourable to them. The histogram obtained in the mixed environment is shown in black and the distribution obtained in the separated environment is show in grey.

In the *mixed* environment, the histogram is centred on the value 0.5 with large variations around its center. In comparison, the histogram in the *separated* environment has always higher bars when the proportion of self-assembled robots on the right side is greater than 0.5. Large gaps are observed between the bars of the histograms, which indicates that more data would be required fore more detailed observations.

The results indicate that in the *mixed* environment, the self-assembled robots do not prefer one side over the other. In the *separated* environment, larger proportions of self-assembled robots are found on the right side of the environment. This indicates an exploitation of the specificities of the *separated* environment i.e. the bias in the environment has promoted the evolution of self-assembly and dis-assembly when required by the environment. However, only some of the self-assembled robots are on the left side. Therefore, the specificity of the environment is not fully exploited.

VI. Conclusion

The results presented in this paper have shown that it is possible to evolve strategies using self-assembly and dis-assembly in the same swarm of robots by relying on a separated environment composed of two sub-environments: one favourable to self-assembly and one favourable to dis-assembly. It has also been shown that a bias in terms of energy given to the self-assembled robots is not sufficient to promote the evolution of self-assembled robots. However, the separation of the environment in sub-environments each favourable to a different strategy is shown to be sufficient. In the separated environment, further analysis of the strategies evolved showed that the self-assembled robots exploit the self-assembly mechanism but not exploit fully the specificities of the environment.

A limitation presented in this paper lies in the exploitation of the separation of the environment: not all self-assembled robots are found in the side that is most favourable to them. To address this challenge, more complex controllers may be needed to exploit the environment. Additionally, other factors such as the design of the self-assembly mechanism can lead to different evolutionary dynamics.

As the main goal of the EER domain lies in the optimization of strategy for real robots, applications to the real world of the effect studied in this paper are envisioned. Future work will investigate such phenomena in situ i.e. on the ChiRP [13] robot, applying wireless charging modules as energy points.

References

[1] R. Gross, M. Bonani, F. Mondada, and M. Dorigo, “Autonomous self-assembly in swarm-bots,” *Robotics, IEEE Transactions on*, vol. 55, no. 6, pp. 1115–1130, 2006.

[2] V. Trianni, E. Tuci, and M. Dorigo, “Evolving functional self-assembling in a swarm of autonomous robots,” *From Animals to Animats*, vol. 8, no. July, pp. 405–414, 2004.

[3] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. Cambridge, MA: MIT Press/Bradford Books, 2000.

[4] C. Ampatzis, E. Tuci, V. Trianni, A. Christensen, and M. Dorigo, “Evolving self-assembly in autonomous homogeneous robots: Experiments with two physical robots,” *Artificial Life*, vol. 15, no. 4, pp. 465–484, Oct. 2009.

[5] R. Gross and M. Dorigo, “Evolution of solitary and group transport behaviors for autonomous robots capable of self-assembling,” *Adaptive Behavior*, vol. 16, no. 5, pp. 285–305, 2008.

[6] E. Tuci, R. Gro, V. Trianni, F. F. Mondada, M. Bonani, and M. Dorigo, “Cooperation through self-assembly in multi-robot systems,” *ACM Transactions on Autonomous and Adaptive Systems*, vol. 1, no. 2, pp. 115–150, 2006.

[7] S. G. Ficici, R. A. Watson, and J. B. Pollack, “Embodied evolution: A response to challenges in evolutionary robotics,” in *Proceedings of the eighth European workshop on learning robots*, 1999, pp. 14–22.

[8] B. Weel, E. Haasdijk, and A. E. Eiben, “The emergence of multi-cellular robot organisms through on-line on-board evolution,” in *Applications of Evolutionary Computation*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7248, no. 7248, pp. 124–134.

[9] B. Weel, M. Hoogendoorn, and A. Eiben, “On-line evolution of controllers for aggregating swarm robots in changing environments,” in *Parallel Problem Solving from Nature-PPSN XII*. Springer Berlin/Heidelberg, 2012, pp. 245–254.

[10] N. Bredeche, J.-M. Montanier, W. Liu, and A. F. T. Winfield, “Environment-driven distributed evolutionary adaptation in a population of autonomous robotic agents,” *Mathematical and Computer Modelling of Dynamical Systems, Special Issue: Modelling the swarm analysing biological and engineered swarm systems*, vol. 18, no. 1, pp. 101–129, 2012.

[11] N. Bredeche, J.-M. Montanier, B. Weel, and E. Haasdijk, “Roborobo! a fast robot simulator for swarm and collective robotics,” *arXiv preprint arXiv:1304.2888*, 2013.

[12] (2014) Roborobo! self-assembly, montanier-jeanmarc.com/files/141209-roborobo-selfassembly.tgz.

[13] C. Skjetne, P. C. Haddow, A. Rye, H. Schei, and J.-M. Montanier, “The ChIRP robot: a versatile swarm robot platform,” in *Robot Intelligence Technology and Applications 2*. Springer International Publishing, 2014, pp. 71–82.