# Genome Variations:
## Effects on the robustness of neuroevolved control for swarm robotics systems

Pedro Romano, Luís Nunes, Anders Lyhne Christensen,
Miguel Duarte, and Sancho Moura Oliveira

Instituto Universitário de Lisboa (ISCTE-IUL)
Instituto de Telecomunicações
Lisboa, Portugal
luis.nunes@iscte.pt

**Abstract.** Manual design of self-organized behavioral control for swarms of robots is a complex task. Neuroevolution has proved a viable alternative given its capacity to automatically synthesize controllers. In this paper, we introduce the concept of Genome Variations (GV) in the neuroevolution of behavioral control for robotic swarms. In an evolutionary setup with GV, a slight mutation is applied to the evolving neural network parameters before they are copied to the robots in a swarm. The genome variation is individual to each robot, thereby generating a slightly heterogeneous swarm. GV represents a novel approach to the evolution of robust behaviors, expected to generate more stable and robust individual controllers, and benefit swarm behaviors that can deal with small heterogeneities in the behavior of other members in the swarm. We conduct experiments using an aggregation task, and compare the evolved solutions to solutions evolved under ideal, noise-free conditions, and to solutions evolved with traditional sensor noise.

**Keywords:** Neuroevolution, robot controllers, genome variations, swarm robotics, robustness, heterogeneity

## 1 Introduction

Synthesizing control for swarms of robots is a challenging process since the individual rules that govern the behavior of each robot are non-trivial to discover and implement [10]. In the research field of evolutionary robotics (ER), neuroevolution (NE) has been successfully applied to the synthesis of self-organized behavioral control for swarms of robots [4]. Given the set of sensor inputs, the controller, an artificial neural network (ANN), determines the value of the outputs that will be sent to the actuators. The weights, and sometimes the topology, of the ANN are evolved by an evolutionary algorithm (EA) to optimize a given fitness function. The evolution of controllers is usually conducted in simulation due to its time-intensive nature, and, after evolution ends, the highest-performing

controllers are transferred to real robotic hardware. Transferring controllers from simulation to real robots often leads to a decrease in performance due to differences between the simulated and real world, an issue known as "the reality gap" [17].

In the evolution of homogeneous swarms, the same controller is cloned for every robot [27], which means that each genome must encode the behavior for all situations and all team members will react similarly in similar conditions. This is different from biological swarms, where each individual has a different genome and slightly different characteristics (physical, behavioral, etc.). The question raised is "do these slight differences in swarm members play a part in the swarms' capabilities to solve problems?". Potter et al. [22] show that behavior homogeneity is a disadvantage when more complex swarm behaviors are needed and specialization becomes a necessity. Even though certain types of ANN-based controllers, such as continuous-time recurrent neural networks (CTRNN) [3], have been shown to enable the emergence of dynamic task-allocation under specific conditions [11], and even in some cases the appearance of different roles [2,23], such controllers tend to display homogeneous behaviors.

This paper introduces the concept of *Genome Variations* (GV), a novel technique that is based on mutating the individual controllers of each robot prior to evaluation. Variations are obtained by adding noise to the weights of the artificial neural networks controlling the robots. The use of GV during training may provide more stable and robust solutions for swarm controllers in coordinated tasks and help bridge the reality gap. There are several potential benefits of varying swarm controllers' genomes during evolution, namely: (i) increased tolerance to minor, but unavoidable, hardware differences between different robots, (ii) increased robustness to environmental noise and to (iii) slight heterogeneities in swarm behavior. Increased tolerance to hardware differences [24], noisy sensors [19] and varying environmental conditions, is expected since a small bias in the input has very similar effects in terms of the internal computations from a change in an input weight or a slight bias in the readings from a sensor. If the GV solution is impervious to small variations in the weights, then it should also be able to tolerate small changes in the input, whether these may be caused by the environment or by the robot's hardware.

GV should also increase robustness to small differences in behaviors of other swarm members, since candidate solutions are exposed to slightly heterogeneous swarms during training. It may be advantageous to have slight behavioral differences in different individuals to break ties that can block the solution to a problem (e.g. two robots both insisting on manipulating the same object). Another way in which GV may be advantageous is by forcing the robots to adapt to different behaviors from teammates, whether these are simply small performance differences or robots experiencing faults.

In this paper, we evaluate the effects of GV in swarms of robots in an aggregation task [26,25]. We chose an aggregation task, since it is a fundamental behavior in many collective systems found in nature [6]. The ability for a group of individuals to aggregate is a precursor to many collective behaviors since be-

ing within sensory range of each other is a paramount condition to coordinate collective swarm behaviors. Also, successful aggregation requires the combined use of different skills that are common with other problems (e.g. distributed search, coordinated movement or cooperation).

We assess the performance of controllers evolved in three different scenarios, an ideal evolutionary scenario with no noise (NF), which is used as a baseline, one with sensor noise (SN), and a Genome Variations scenario. The highest performing solution evolved in each scenario is tested in all three setups (NF, SN and GV).

## 2 Related Work

Variation is a fundamental part of the evolutionary process, since new solutions are based on previously successful solutions by applying evolutionary operators such as mutation and recombination [1]. However, the use of variation for other purposes in the evolution of controllers for swarms of robots remains unexplored. In this section, we present a discussion of work related to variation, robustness and generation of heterogeneity in swarm robotics.

The need to create robust solutions in simulation was identified early on in the field of evolutionary robotics (ER). Cliff et al. [8] use the lowest fitness score obtained, as opposed to the mean, obtained in multiple evaluation samples of the evolved individuals in a simulated visually-guided robot navigation problem. The authors adopt the selection method "to encourage robustness, remembering that there is noise at all levels in the system". Miglino et al. [19] and later Jakobi [17] build on the ideas in [8] and apply noise to simulated sensors, actuators, and environmental elements to create controllers that are more capable of crossing the reality gap.

As other authors attempted alternatives to improve solution robustness that did not involve noise, to compensate for the overhead that noisy conditions impose on training in simulation [21], Gomez and Miikkulainen [16] report that "the appropriate use of noise during evolution can improve transfer [of behaviors to real world scenarios] significantly by compensating for inaccuracy in the simulator". The "appropriate use" refers to adding sensor noise and trajectory noise. Authors report significant improvements in transferred behavior performance.

Lehman et al. [18] introduced the concept of *reactivity* and showed that it is as effective in addressing the robustness problem as training with noise, while being computationally more efficient. Reactivity promotes the learning agent to use behaviors where there are dependencies in the magnitude of changes in a robot's sensors and actuators, i.e. if there is a large change in the sensors, there should be some large change in the actuators. Reactivity should not be confused with the concept of a *reactive agent* usually associated with the subsumption architecture [5].

In the evolution of heterogeneous multirobot teams and swarms, either co-evolution [22,14], or approaches in which whole teams are encoded in a single genome [22] are typically used. Potter et al. [22] show that as tasks be-

come more difficult, heterogeneity and specialization become more important, although there is a trade-off in learning speed. Nitschke et al. [20] study the evolution of collective behaviors, with a focus on facilitating specialization and team heterogeneity. The authors use genotypic and behavioral difference metrics to group genomes in different sub-populations in order to regulate inter-population recombination. Gomes et al. [14] propose Hyb-CCEA to facilitate the evolution of heterogeneous, cooperative multi-agent systems using co-evolution and a method for merging and splitting sub-teams.

Silva et al. [25] apply online evolution [13] to swarms of robots by continuously generating controllers that can adapt to periodic changes during task execution. Shang et al. [24] report a variant of EA for creating heterogeneity by adapting to each robot's specific hardware: robots compensate for hardware differences by individually retraining to adapt to the specificities of their hardware before re-insertion in the team. D'Ambrosio et al. [9] see each element of the team as a combination of policies, creating a continuum between homogeneity and heterogeneity. As in our approach, all agents have the same base-genome, but these agents also possess a "policy geometry" that makes the behavior of the agent depend on its position and provides for the necessary diversity, as well as keeping the homogeneity of the basic genome.

In the work discussed above, no attempt was made to generate heterogeneity in team or swarm behavior simply by introducing genome variations before evaluation. The most common approach to achieve robustness is by introducing noise during training at different levels of the simulation (sensors, actuators, etc.). Also, the focus of heterogeneity is on task specialization, which generally requires separate evolution of individuals or sub-teams, not on the effects of small genome heterogeneities, as seen in biological swarms and in our approach.

## 3 Methodology

Artificial evolution is typically an iterative process composed of different stages. The stages can be divided into: population generation, genome evaluation and genome selection. Iterations are usually referred to as *generations*. The first generation is commonly composed of a number of randomly-generated genomes. During evaluation, the fitness of each genome is typically sampled several times under different initial conditions to get a proper estimate of the genome's quality. In each sample, the genome is decoded into a controller that is copied to each robot, and the performance of the swarm with respect to the target task is assessed. In the final stage of the iterative process, a set of genomes is selected for reproduction based on the fitness scores obtained. The selected genomes are used as the seed for the next generation. The process typically continues for a fixed number of generations, until stagnation is detected, or until a solution of sufficient quality has been evolved.

In this study, the robotic controllers that are decoded from genomes are CTRNNs [3]. Genomes are composed of the values of the weights, bias and recurrent connections of a fixed-size CTRNN (in this case: 8 x 5 x 2, with delayed

self-connections at each node). In the evaluation phase, the fitness of each genome is sampled 30 times in order to assess its fitness. The final fitness score corresponds to the mean fitness score obtained in the 30 samples. After all genomes of a generation have been evaluated, we use a simple elitist selection strategy that retains the five highest-performing controllers from a generation and also uses these elite genomes to create the new genomes for the next generation (a total of 100 genomes). A new genome is created by mutating one of the elite genomes. The evolutionary process continues until the desired number of generations is reached (400).

In the case of conventional evolution with ANN-based controllers for swarms of robots, an individual's genome is decoded into an ANN, which is copied to each robot in the swarm. With the GV approach studied in this paper, the genome assigned to each robot in the swarm prior to evaluation is not an exact clone of the genome under evolution. Instead, a slight variation of the base-genome is copied to each robot. GV is applied to the base-genome in each evaluation sample by adding a uniformly distributed random value (noise) in a defined range, to a certain portion of the ANN weights, creating an heterogeneous swarm. Different ranges for both the variation magnitude and portion of weights varied are tested on the experiments conducted in this study (see Section 4). GV is only used when assessing the performance of a genome. A different random variation is applied to the ANN copied to each robot, thereby generating a different heterogeneous swarm in every sample from the same base genome.

For our experiments, we use JBotEvolver [12], an open-source simulation platform and neuroevolution framework.

## 4 Experiments

In this section, we compare our GV approach, neuroevolution with (SN) and without sensor noise (NF). Adding noise to simulated sensors during evolution has been shown to promote robust behaviors that are more capable of crossing "the reality gap" [19], so we will replicate evolution with sensor noise to compare with the robustness of the GV evolution in different scenarios. In the following experiments, we study GV with different settings for both the magnitude of variations and the proportion of ANN parameters varied.

The task used in these experiments is standard swarm robotics aggregation [26]. Our aggregation task is performed in a bounded arena, where 15 robots should find each other and aggregate, i.e. form a connected cluster with the greatest number of robots possible. A robot is considered in a cluster if it is within 0.25 m of at least one other robot from that cluster.

Robots can sense each other with four sensors dispersed evenly on their body, and can also sense walls, using four sensors that are also dispersed evenly on their body. All sensors have an opening angle of $90^{\text{o}}$, providing each robot with omni-directional sensing, and they have a range of 1 m. All sensory readings are linearized to the interval $[0, 1]$ before being fed to the controller. The value of the actual reading of the sensor is linearly proportional to the distance at which the

object is sensed. In this task, the size of the arena is 4 m in odd samples and 8 m in even samples. The size of the arena is varied to prevent overfitting of solutions to a particular size. Controllers are scored based on the largest cluster of robots in the environment, according to the following equation inspired by [15]:

$$Fitness = \frac{\sum_{t=0}^{T} \left( max \left( \sum_{r=0}^{R} \left( C_{rt} \right) \right)^2 \right)}{T} \quad (1)$$

where $T$ is the number of control cycles in the sample (3000), $R$ the number of robots (15), and $C_{rt}$ is the number of robots clustered with robot $r$ at control cycle $t$.

Throughout the following experiments we have three scenarios: Noise-Free (NF), Sensor Noise (SN), and Genome Variations (GV), that will be detailed in Sections 4.1, 4.2 and 4.3, respectively. We assess the performance of genomes evolved in each scenario in all test setups (NF, SN and GV).

The noise-free NF scenario will be the baseline. In the SN scenario (Section 4.2), each sensor in each robot is assigned with an offset uniformly distributed within the range $[-0.1, 0.1]$, and a random noise on each of its sensor readings, also in the same range. Sensor offset noise is kept constant through the sample's lifetime while sensor reading noise is random for each reading. Total noise in the SN environment has an amplitude lower than the 0.4 suggested by [17]. In the SN scenario, all genomes are evolved in a noisy environment. In the GV scenario (Section 4.3), GV is used during evolution. We study three different variants of the GV scenario, each with different parameters: (i) $[-0.125, 0.125]$ variation on 12.5% of the ANN weights, (ii) $[-0.25, 0.25]$ variation on 25% of the ANN weights, iii) $[-0.5, 0.5]$ variation on 50% of the ANN weights. Post-evolution we assess the performance of the highest scoring controllers from each of the scenarios in all three test setups: NF, SN, and GV.

We are interested in assessing how the different solutions perform, and in comparing the robustness and self-organization capabilities of controllers evolved in the different scenarios. Five evolutionary runs are conducted in each scenario (NF, SN, and three GV variants) with a total of 400 generations per run. Each generation is composed of 100 genomes, which are evaluated in 30 samples each. After the evolutionary process ends, the highest-performing genome of the last generation is post-evaluated in 100 samples in the different test scenarios considered for each experiment.

### 4.1 Noise-free Neuroevolution

The first set of experiments serves as a baseline: we conducted a total of five evolutionary runs in the noise-free (NF) scenario. We then assessed the performance of the highest-scoring genome evolved in each run in all test setups, see results in Figure 1. Notice that swarms are not being evolved with SN or GV, only in the post-evolution assessment is SN or GV introduced.

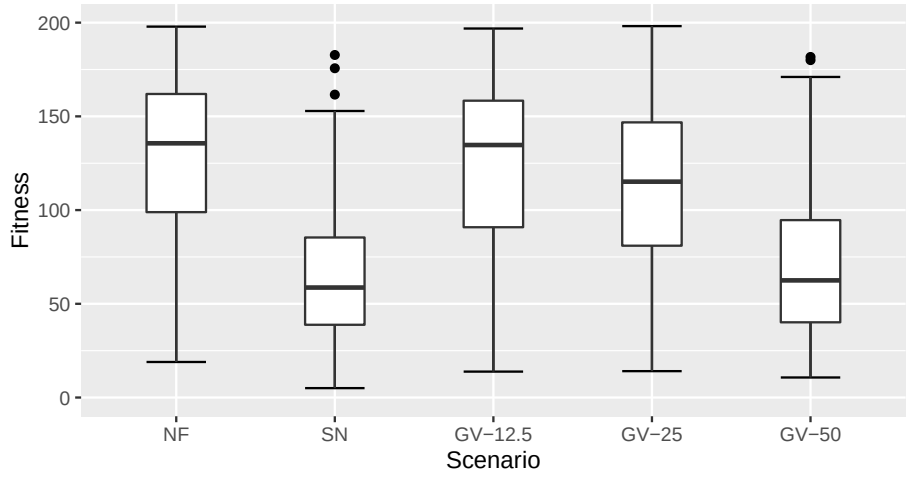


**Fig. 1.** Results for NF evolved controller in the three different scenarios: NF, SN and GV with $[-0.125, 0.125]$ variation on 12.5% of the ANN weights (GV-12.5), GV with $[-0.25, 0.25]$ variation on 25% of the ANN weights (GV-25) and GV with $[-0.5, 0.5]$ variation on 50% of the ANN weights (GV-50). Mean fitness obtained in 100 post-evaluation trials with the highest-scoring controller for each run (five runs), resulting in a total of 500 observations per box.

Controllers evolved in the NF scenario suffer from a significant fitness drop when exposed to SN. Recall that in the GV test setups, we use a different variation for each robot in the swarm, creating an heterogeneous swarm. Still, since controllers were not evolved with GV, performance degrades as the GV interval increases (GV-12.5, GV-25 and GV-50). This provides an indication of the amount of variation the solutions can cope with, before performance degrades, namely between 0.25 and 0.5, in GV-25 and GV-50.

### 4.2 Sensor Noise Environment

In the second set of experiments, we conducted five evolutionary runs in the SN scenario, and assessed the performance of the highest-scoring genome evolved in each run in all test setups, see results in Figure 2. Details on how SN is applied can be found in the beginning Section 4.

Controllers evolved in a SN environment have a good behavior in both NF and SN test setup. The GV used in this test (heterogeneous swarms) maintains a performance on par with NF and SN trained up to GV-25. Performance in tests with high GV intervals (GV-50) degrades but performance is still higher than the noise-free-evolved solutions in the SN test (SN in Figure 1). It can be expected since SN evolution should enable the genome to cope with noise in the input up to $[-0.1, 0.1] \pm [-0.1, 0.1]$.

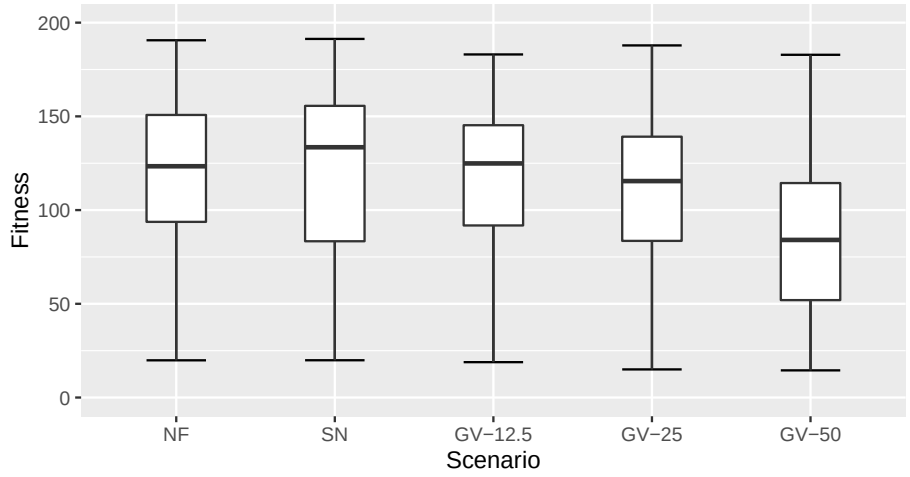


**Fig. 2.** Results for SN evolved controller in the three different scenarios: NF, SN and GV. In this setup noise was added to random sensors in each robot. The GV scenarios have $[-0.125, 0.125]$ variation on 12.5% of the ANN weights (GV-12.5), GV with $[-0.25, 0.25]$ variation on 25% of the ANN weights (GV-25) and GV with $[-0.5, 0.5]$ variation on 50% of the ANN weights (GV-50). Mean fitness obtained in 100 post-evaluation trials with the highest-scoring controller for each run (five runs), resulting in a total of 500 observations per box.

In an additional set of experiments (not shown), GV evolved controllers were tested in a setup that had both SN and GV simultaneously. These experiments tested homogeneous versus heterogeneous swarms in SN scenarios, but we chose not to include them since they were not significantly different from the results presented in Figure 2.

### 4.3 Genome Variations

In the third set of experiments, we conducted five evolutionary runs in each of the three GV scenarios, and assessed the performance of the highest-scoring genome evolved in each run in all test setups, see results in Figure 3.

When using a small variation interval (GV-12.5), GV evolved solutions are similar to the solutions evolved in the NF scenario. In this set of experiments, both NF and GV-12.5, evolved behaviors in which the robots aggregate with search-like patterns moving freely inside the environment. GV assessments with larger variation intervals, GV-25 and GV-50, sometimes evolved a behavior in which the robots group by following the walls, turning corners, and eventually finding each other. All the behaviors evolved in the GV scenario solved the aggregation task in the NF test setup.

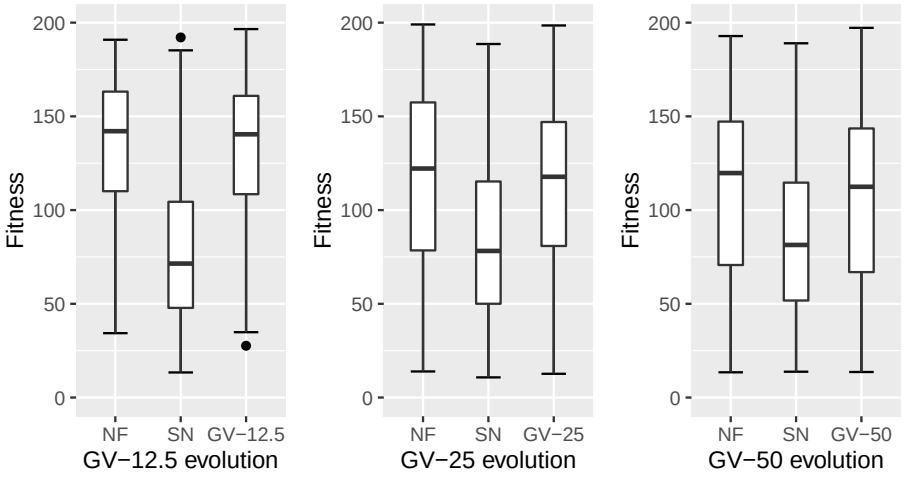


**Fig. 3.** Results for GV-12.5, GV-25 and GV-50 evolved controller in the three different scenarios: NF, SN and GV (heterogeneous swarms). In GV tests, the variation interval is the same as the one used on the evolution of that specific controller. Mean fitness obtained in 100 post-evaluation trials with the highest-scoring controller for each run (five runs), resulting in a total of 500 observations per box.

While a $[-0.125, 0.125]$ variation on 12.5% of the weights has almost no impact in the fitness, larger variations in more weights lead to solutions with an increasingly worse fitness. GV-evolved solutions generally required more time to form a group (around 1700 control cycles on NF evolved controllers and 2040 on GV-25 — tests done in the NF test scenario, small arena). Apart from the quantitative evaluation shown above, a qualitative evaluation of the solutions shows that the solutions evolved with GV with a low degree of variation, appear, to the naked eye, to be equal to the ones evolved in the noise-free NF setup. Also, robots in GV-evolved solutions with larger variation intervals lack the smoothness of movements that NF and narrow-variation GV's have.

As seen in Section 4.1, when tested in SN setup, controllers evolved in the NF scenario have a significant drop in fitness when exposed to SN. Surprisingly, the fitness drop of GV-evolved controllers in presence of SN is comparable to that of noise-free-evolved controllers in this case. As mentioned above, larger GV intervals tend to evolve different behaviors. The behavior where the robots aggregate by following the walls and finding each other near them is probably a more robust strategy in a SN environment.

## 5 Conclusions and Future Work

In this paper, we introduced Genome Variations, a technique in which small mutations are applied to the evolving neural networks before they are copied to

each robot in a swarm. The variation is introduced only in the evaluation of the swarm, and it is individual to each robot, leading to slight behavioral heterogeneity in the swarm. GV was expected to provide a certain degree of robustness whether by resembling the effects of sensor noise, and/or by promoting behaviors robust enough to cope with slightly different peer behaviors during evolution.

Experiments in the aggregation task only partially confirm the hypothesis. GV does indeed appear to produce more robust behaviors than those evolved in noise-free scenarios, but robustness is not comparable to solutions evolved with sensor noise, nor does the resulting mild degree of heterogeneity seem to have any significant impact on the robots' behavior in this task.

Future work will focus on increasing the number of evolutionary runs and include additional tasks to provide a sufficient basis to determine the significance of any performance differences. We intend to determine whether GV can provide some form of generic robustness that is less expensive to train than tolerance to each specific type of noise applied to sensors, actuators, or to the environment. We will also try to use NEAT (more reactive and flexible than CTRNNs). NEAT will also allow an interesting analysis of the different types of networks evolved with each setup (number of nodes, connections, recurrent connections). Also, we plan to assess the impact of GV on swarm fault-tolerance and its potential advantages on behavior transfer from simulation to real-robot scenarios.

## Acknowledgements

This work was supported by Fundação para a Ciência e a Tecnologia (FCT) under the grants, SFRH/BD/76438/2011, UID/EEA/50008/2013, and EXPL/EEI-AUT/0329/2013, integrated in the CORATAM and HANCAD projects [7].

## References

1. Bäck, T., Schwefel, H.P.: An overview of evolutionary algorithms for parameter optimization. Evolutionary computation 1(1), 1–23 (1993)
2. Baldassarre, G., Nolfi, S., Parisi, D.: Evolving mobile robots able to display collective behaviors. Artificial life 9(3), 255–267 (2003)
3. Beer, R.D., Gallagher, J.C.: Evolving Dynamical Neural Networks for Adaptive Behavior. Adaptive Behavior 1(1), 91–122 (1992)
4. Bongard, J.C.: Evolutionary robotics. Communications of the ACM 56(8), 74–83 (2013)
5. Brooks, R.A.: Elephants don't play chess. Robotics and autonomous systems 6(1), 3–15 (1990)
6. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-organisation in biological systems. Princeton University Press (2001)
7. Christensen, A.L., Oliveira, S., Postolache, O., João de Oliveira, M., Sargento, S., Santana, P., Nunes, L., Velez, F., Sebastião, P., Costa, V., Duarte, M., Gomes, J., Rodrigues, T., Silva, F.: Design of Communication and Control for Swarms of Aquatic Surface Drones. In: Proceedings of the International Conference on Agents and Artificial Intelligence (ICAART). pp. 548–555. SCITEPRESS, Lisbon, Portugal (2015)

8. Cliff, D., Husbands, P., Harvey, I.: Evolving visually guided robots. In: Proceedings of the Second International Conference on Simulation of Adaptive Behavior (SAB). pp. 374–383. MIT Press, Cambridge, MA (1993)
9. D'Ambrosio, D.B., Stanley, K.O.: Scalable multiagent learning through indirect encoding of policy geometry. Evolutionary Intelligence 6(1), 1–26 (2013)
10. Dorigo, M., Trianni, V., Şahin, E., Groß, R., Labella, T., Baldassarre, G., Nolfi, S., Deneubourg, J., Mondada, F., Floreano, D., Gambardella, L.M.: Evolving self-organizing behaviors for a swarm-bot. Autonomous Robots 17(2), 223–245 (2004)
11. Duarte, M., Oliveira, S., Christensen, A.L.: Towards Artificial Evolution of Complex Behavior Observed in Insect Colonies. In: Proceedings of the Portuguese Conference on Artificial Intelligence (EPIA). pp. 153–167. Springer, Berlin, Germany (2011)
12. Duarte, M., Silva, F., Rodrigues, T., Oliveira, S.M., Christensen, A.L.: JBotEvolver: A versatile simulation platform for evolutionary robotics. In: Proceedings of the International Conference on the Synthesis & Simulation of Living Systems (ALIFE). pp. 210–211. MIT Press, Cambridge, MA (2014)
13. Ficici, S.G., Watson, R.A., Pollack, J.B.: Embodied evolution: A response to challenges in evolutionary robotics. In: Proceedings of the European workshop on learning robots. pp. 14–22. Citeseer (1999)
14. Gomes, J., Mariano, P., Christensen, A.L.: Cooperative Coevolution of Partially Heterogeneous Multiagent Systems. In: Proceedings of the International Conference on Autonomous Agents and Multiagent Systems. pp. 297–305. International Foundation for Autonomous Agents and Multiagent Systems (2015)
15. Gomes, J., Urbano, P., Christensen, A.L.: Evolution of swarm robotics systems with novelty search. Swarm Intelligence 7(2-3), 115–144 (2013)
16. Gomez, F.J., Miikkulainen, R.: Transfer of neuroevolved controllers in unstable domains. In: Proceedings of Genetic and Evolutionary Computation Conference (GECCO). pp. 957–968. Springer, Berlin, Germany (2004)
17. Jakobi, N.: Evolutionary robotics and the radical envelope-of-noise hypothesis. Adaptive behavior 6(2), 325–368 (1997)
18. Lehman, J., Risi, S., D'Ambrosio, D.B., Stanley, K.O.: Rewarding reactivity to evolve robust controllers without multiple trials or noise. In: Proceedings of the Thirteenth International Conference on Artificial Life (ALIFE). pp. 379–386. MIT Press, Cambridge, MA (2012)
19. Miglino, O., Lund, H.H., Nolfi, S.: Evolving mobile robots in simulated and real environments. Artificial Life 2(4), 417–434 (1995)
20. Nitschke, G.S., Eiben, A.E., Schut, M.C.: Evolving team behaviors with specialization. Genetic Programming and Evolvable Machines 13(4), 493–536 (2012)
21. Paenke, I., Branke, J., Jin, Y.: Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation. IEEE Transactions on Evolutionary Computation 10(4), 405–420 (2006)
22. Potter, M.A., Meeden, L.A., Schultz, A.C.: Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI). pp. 1337–1343. Citeseer (2001)
23. Quinn, M., Smith, L., Mayley, G., Husbands, P.: Evolving controllers for a homogeneous system of physical robots: structured cooperation with minimal sensors. Philosophical transactions. Series A, Mathematical, physical, and engineering sciences 361(1811), 2321–2343 (2003)

24. Shang, B., Crowder, R., Zauner, K.P.: Simulation of Hardware Variations in Swarm Robots. In: Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics. pp. 4066–4071. IEEE Press, Piscataway, NJ (2013)
25. Silva, F., Urbano, P., Correia, L., Christensen, A.L.: odNEAT: An Algorithm for Decentralised Online Evolution of Robotic Controllers. Evolutionary Computation 23(3), 421–449 (2015)
26. Trianni, V., Nolfi, S., Dorigo, M.: Cooperative hole avoidance in a swarm-bot. Robotics and Autonomous Systems 54(2), 97–103 (2006)
27. Waibel, M., Keller, L., Floreano, D.: Genetic team composition and level of selection in the evolution of cooperation. IEEE Transactions on Evolutionary Computation 13(3), 648–660 (2009)