Nat Comput (2019) 18:29–46
https://doi.org/10.1007/s11047-016-9582-1

Challenges in cooperative coevolution of physically heterogeneous robot teams

Jorge Gomes[1,2,3] · **Pedro Mariano**[3] · **Anders Lyhne Christensen**[1,2,4]

Published online: 9 September 2016

Abstract Heterogeneous multirobot systems have shown significant potential in many applications. Cooperative coevolutionary algorithms (CCEAs) represent a promising approach to synthesise controllers for such systems, as they can evolve multiple co-adapted components. Although CCEAs allow for an arbitrary level of team heterogeneity, in previous works heterogeneity is typically only addressed at the behavioural level. In this paper, we study the use of CCEAs to evolve control for a heterogeneous multirobot system where the robots have disparate morphologies and capabilities. Our experiments rely on a simulated task where a simple ground robot must cooperate with a complex aerial robot to find and collect items. We first show that CCEAs can evolve successful controllers for physically heterogeneous teams, but find that differences in the complexity of the skills the robots need to learn can impair CCEAs' effectiveness. We then study how different populations can use different evolutionary algorithms and parameters tuned to the agents' complexity. Finally, we demonstrate how CCEAs' effectiveness can be improved using incremental evolution or novelty-driven coevolution. Our study shows that, despite its limitations, coevolution is a viable approach for synthesising control for morphologically heterogeneous systems.

✉ Jorge Gomes
jgomes@di.fc.ul.pt

[1] BioMachines Lab, Lisbon, Portugal

[2] Instituto de Telecomunicações, Lisbon, Portugal

[3] BioISI, Faculdade de Ciências, Universidade de Lisboa, Lisbon, Portugal

[4] Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon, Portugal

Keywords Evolutionary robotics · Cooperative coevolution · Multirobot systems · Heterogeneous systems · Premature convergence · Artificial neural networks

1 Introduction

Cooperative coevolutionary algorithms (CCEAs) have been advocated as a valuable approach for the evolution of heterogeneous multiagent systems (Potter et al. 2001). In the classic CCEA architecture (Potter and Jong 2000), each agent evolves in an isolated population, and the individuals from a population are evaluated by forming collaborations with individuals from the other populations. One key advantage of CCEAs is that since populations are isolated, it is possible for different populations to evolve radically different agents, with different genomes, and even to use different evolutionary algorithms. This possibility of arbitrary heterogeneity has, however, only been exploited to a limited extent. Most previous works focus on the evolution of controllers for behaviourally heterogeneous, but morphologically homogeneous, multiagent systems (see for instance Nitschke et al. 2012b; Potter et al. 2001; Yong and Miikkulainen 2009). This means that all agents in the system have a similar complexity, similar capabilities, and use the same genotype representation.

Morphologically heterogeneous multirobot systems have shown significant potential in a number of applications (Dorigo et al. 2013; Duan and Liu 2010; Howard et al. 2006). The cooperation between morphologically heterogeneous robots can, for instance, enable the achievement of tasks that are beyond the reach of a single type of robot. The behavioural control for morphologically heterogeneous systems is typically designed manually.

This process can, however, be challenging, as behavioural control must integrate the capabilities of different robot types in a way that the efficiency of the complete system becomes greater than if the different robot types worked independently without cooperation (Dorigo et al. 2013).

In this paper, we study how cooperative coevolutionary algorithms can be used to evolve effective controllers for agents with radically different capabilities, in a task that requires tight cooperation. Cooperative coevolution is traditionally associated with a number of challenges (Wiegand 2003) that stem from the intricate dynamics of coevolving two or more populations, where the evaluation of the individuals in one population depends on the individuals of the other populations. A key element in the evolution of cooperative behaviours is *synchronised learning* (Uchibe et al. 1998): populations should exhibit a mutual development of skills, in order to avoid loss of fitness gradients and convergence to mediocre stable states. We will study how a high degree of heterogeneity in a multirobot system affects the mutual development of skills.

Our experiments are based on a simulated item collection task, where a ground robot with very limited capabilities must cooperate with an aerial robot with a significantly more complex sensor-effector configuration. We explore several task variants to study how the differences between the robots, regarding the skills that must be evolved, and the difficulty in achieving cooperation, affect the performance of cooperative coevolution. Next, we study how cooperative coevolution performs when faced with an additional level of heterogeneity in the populations: the use of different evolutionary algorithms in each of the populations. Finally, we try to improve the effectiveness of coevolution with three techniques found in previous works: incremental evolution (Gomez and Miikkulainen 1997), multi-objectivisation of sub-goals (Mouret and Doncieux 2008), and novelty-driven cooperative coevolution (Gomes et al. 2014a, 2016b). We compare the advantages and drawbacks of each technique, and study how they can contribute to the effective coevolution of behaviours for highly heterogeneous multirobot systems.

This paper is an extension of the work reported in (Gomes et al. 2015a). Here, we use a more realistic task setup where the initial positions of the robots are randomised, thus requiring the evolution of more general behaviours. We provide a richer insight into the challenges in coevolving behaviours for heterogeneous systems, supported by an analysis of team behaviour, and we study heterogeneity at the algorithmic level by using different evolutionary algorithms and different parameters in the different populations. Novelty search is now implemented as part of a multiobjective evolutionary algorithm (MOEA) following recent results (Gomes et al. 2016b), and we also use the MOEA to implement a form of incremental evolution. Finally, we provide a thorough discussion on the challenges associated with coevolving highly heterogeneous multiagent teams, and propose a number of guidelines for these challenges.

2 Related work

2.1 Morphologically heterogeneous systems

Heterogeneous multirobot systems are characterised by the morphological and/or behavioural diversity of their constituent robots. Behavioural heterogeneity is commonly employed to allow behaviour specialisation (Nitschke et al. 2012b; Potter et al. 2001). In morphologically heterogeneous systems, on the other hand, robots have different actuation and sensing capabilities, and collaborate to take advantage of the collective set of capabilities (Dorigo et al. 2013).

The Swarmanoid project (Dorigo et al. 2013) studied morphologically heterogeneous robotic swarms: groups of aerial, ground, and climbing robots, with decentralised control, to operate in three-dimensional human-centric environments. Some of the studied tasks include: a gap crossing task where ground robots receive instructions from aerial robots (Mathews et al. 2010); an indoor navigation task where aerial robots aid ground robots in cluttered environments (Ducatelle et al. 2011); and a search-and-retrieval task in a 3-D environment, where all robot types cooperate (Dorigo et al. 2013). Other works outside Swarmanoid have also shown the potential of cooperation between ground and aerial robots, especially in search-and-rescue tasks (Duan and Liu 2010). Aerial robots have a privileged perspective of the environment, and can therefore be used to assist ground robots in a variety of tasks (Hsieh et al. 2007; Lacroix and Le Besnerais 2011).

Morphologically heterogeneous systems also encompass systems composed of robots of a similar nature (e.g., ground robots only). Heterogeneity can be used to reduce the cost of the system, by assigning different sensor/actuator capabilities to different robots, which can then cooperate to take advantage of each other's capabilities. Grabowski et al. (2000) showed such an approach in a mapping and exploration task, using multiple types of ground robots equipped with complementary sensors. In (Parker et al. 2004), capable leader robots assist sensor-limited robots in navigating indoor environments. Howard et al. (2006) extended this approach to a task where few complex robots cooperate with a large number of inexpensive robots to map the environment and establish a sensor network.

In some cases, morphological differences in a multirobot system are not a choice, but rather a practical

constraint. A multirobot system might be composed of the different types of robots that are currently available to perform the task (Blumenthal and Parker 2004; Candea et al. 2001; Jones et al. 2006). In these situations, behavioural control must take the differences in robot capabilities into account.

2.2 Cooperative coevolution

In the studies on multirobot systems discussed above, distributed control was achieved by manually designing the behavioural rules of the individual robots. Previous works have shown that this can be a challenging task, since the decomposition of the desired global behaviour into individual behavioural rules is often complex and inconspicuous (Dorigo et al. 2004). This challenge is exacerbated in heterogeneous systems (Dorigo et al. 2013), as behavioural control must be able to integrate the different abilities of different robot types to work in synergy towards the achievement of a common goal. One possible solution to this problem is the use of evolutionary algorithms to synthesise robot controllers (Dorigo et al. 2004; Potter et al. 2001; Uchibe et al. 1998). Besides automating the controller design, evolutionary algorithms have the potential to discover optimal solutions for the problem, and to discover diverse, unexpected solutions (Nolfi and Floreano 2000).

Cooperative coevolutionary algorithms are a natural fit for the evolution of heterogeneous multiagent systems (Potter et al. 2001). The classic cooperative coevolution architecture (Potter and Jong 2000) operates with a system comprising two or more populations, where each agent evolves in a separate population. Each population is typically isolated, meaning that individuals only compete and reproduce with members of their own population. The individuals of a population are evaluated by forming teams with representative individuals from the other populations, where the individual under evaluation is assigned the fitness score obtained by the team as a whole. The fitness gradient is therefore relative: it is strictly a function of the individuals' contribution within the context of the other populations.

One advantage of CCEAs is that, due to the separation of populations, an arbitrary level of heterogeneity can be accommodated within the system. Cooperative coevolution, however, is typically applied to morphologically homogeneous systems, focusing only on behavioural specialisation. Some examples include the *predator-prey* task (Gomes et al. 2014a; Nitschke et al. 2012a; Yong and Miikkulainen 2009); the *herding* task (Gomes et al. 2015b; Potter et al. 2001); the *gathering and collective construction* task (Nitschke et al. 2012b); the *multi-rover* task (Nitschke et al. 2009); and *keepaway soccer* (Gomes et al. 2014a). There have only been few reports of successful evolution of morphologically heterogeneous systems, and in these studies, agents had only minor morphological differences, for instance: a *keepaway soccer* task where agents have different moving and passing speeds (Gomes et al. 2014a); a *foraging* task where agents have different movement speed and sensing ranges (Yang et al. 2012); a *predator-prey* task where the predators have slightly different linear and turning speeds (Blumenthal and Parker 2004); and a *cooperative foraging* task where there are two different robot types with the same capabilities (Knudson and Tumer 2010).

2.3 Overcoming challenges in coevolution

In a CCEA, the search space of each population is constrained by the individuals in the other populations. The search space is thus constantly changing, and the fitness of an individual can vary significantly depending on the collaborators that constitute the rest of the team. It is therefore easy for a population to be misled by the other populations. This evolutionary dynamic can cause two known pathologies: convergence to mediocre stable states (Panait 2010) and loss of fitness gradients (Wiegand 2003).

Convergence to mediocre stable states occurs when populations are unable to further improve their individuals, given the current set of collaborators drawn from the other populations. Previous works have shown that CCEAs tend to gravitate towards equilibrium states, not necessarily optimal solutions, which can impair their effectiveness (Gomes et al. 2014a, 2016b; Panait 2010). Loss of fitness gradient is more common in competitive coevolution, but can also appear in CCEAs: it occurs when a population reaches a state such that the other populations lose the fitness diversity necessary for meaningful progress (Wiegand 2003).

We hypothesise that when populations have to evolve substantially different skills, with different complexity, synchronised learning is less likely to occur, causing convergence to mediocre stable states and/or loss of fitness diversity. In this paper, we study this issue and how it can be mitigated. To this end, we evaluate incremental evolution (Gomez and Miikkulainen 1997), multi-objectivisation with sub-objectives (Mouret and Doncieux 2008; Trianni and López-Ibáñez 2015), and novelty search (Gomes et al. 2014a, 2016b; Lehman and Stanley 2011) as means to improve coevolution's effectiveness.

2.3.1 Task decomposition

One common approach to avoid premature convergence in evolutionary robotics consists of decomposing the larger goal in sub-goals that are easier to achieve (Doncieux and

Mouret 2014). In an incremental evolution scheme (Gomez and Miikkulainen 1997), a series of evolutionary stages are defined by the experimenter, and evolution moves from one stage to the next when the population reaches a *sufficient* performance level. Previous works have shown that incremental evolution schemes, such as environmental complexification (Christensen and Dorigo 2006; Gomez and Miikkulainen 1997), can be successfully used to assist cooperative coevolution (Nitschke et al. 2012b; Yong and Miikkulainen 2009). In an environmental complexification setup, solutions are initially evaluated in a simplified version of the environment, which becomes progressively more complex as the evolutionary process is able to find solutions for the current stage.

Another form of incremental evolution, used in our study, is staged evolution (Doncieux and Mouret 2014). In staged evolution, the environment is the same throughout evolution, but the objectives change. The fitness function typically rewards simple objectives earlier in evolution, and more complex objectives later on. Uchibe and Asada (2006) show how staged evolution can also be used to integrate cooperative and competitive coevolution in a multirobot system.

While the approaches discussed above facilitate the evolution of complex behaviors, they require in-depth knowledge of the global task and how it can be divided into sub-tasks. Mouret and Doncieux (2008) proposed an alternative that reduces the dependency on the experimenter's knowledge: all the sub-objectives are optimised at the same time using a multiobjective evolutionary algorithm (MOEA). Additional details can be found in a survey of the potential of MOEAs in evolutionary robotics by Trianni and López-Ibáñez (2015).

2.3.2 *Novelty search*

A different approach to avoid premature convergence in evolutionary robotics is by adopting a more open-ended and exploratory evolutionary process. Novelty search (Lehman and Stanley 2011), for instance, is a widely recognised approach for overcoming premature convergence in evolutionary robotics applications (Gomes et al. 2015c; Mouret and Doncieux 2012). The idea behind novelty search is to create a divergent evolutionary process by rewarding individuals that display novel behaviours, instead of rewarding the most fit individuals according to a given fitness function. Novelty search is, however, commonly combined with fitness-driven evolution to balance exploration and exploitation in the evolutionary process (Gomes et al. 2015b).

Gomes et al. (2014a, 2016b) proposed a novelty-based method for avoiding convergence to equilibrium states in cooperative coevolution. The proposed approach (*NS-T*) relies on team-level behaviour characterisations to reward behaviourally novel collaborations, in addition to high-fitness ones. The team-level characterisations capture how the team as a whole behaves, without discriminating between the behaviours of the individual agents. Gomes et al. (2016b) showed that *NS-T* can reach collaborations associated with higher fitness scores more often than a traditional CCEA, and can evolve a diverse set of solutions for a given task in a single evolutionary run. The advantages of novelty-driven coevolution have been shown in a number of tasks (Gomes et al. 2016b), and have been validated in a real multirobot system (Gomes et al. 2016a).

3 Cooperative item collection task

We use a cooperative item collection task to study coevolution in highly heterogeneous multirobot systems. In this task, an aerial robot must assist a ground robot in collecting items randomly dispersed in an unbounded environment, see Fig. 1. The ground robot has significantly fewer sensory capabilities than the aerial robot (detailed below). The two robots cannot communicate explicitly: they can only sense the relative position of each other when in close proximity. To accomplish the task, the aerial robot must learn to find the ground robot, and then guide it towards collectable items in the environment. Complementary, the ground robot should follow the aerial robot and collect the items found.

3.1 Robot configurations

Each robot is independently controlled by a neural network, which takes normalised sensor readings as inputs and its outputs control the robot's actuators. Both robots have sensors modelled after a vertical camera sensor, facing up and down for the ground robot and aerial robot, respectively. The camera has a field of view of 60°, and can detect objects up to a vertical distance of 250 cm, see Fig. 1. The camera image is not directly used by the robots: the sensor cone is divided in equal-sized sectors, and the value of each input is the presence or distance of the object in the respective circle sector (see Fig. 2). The robots rely on these camera-based inputs for detecting each other. In the case of the aerial robot, camera-based inputs are also used to detect the items that must be collected. The sensory inputs available to each robot are listed in Table 1.

Only the ground robot has the ability to collect items from the environment. To collect an item, the robot simply has to pass over it. The ground robot is a small (8 cm diameter) differential drive robot. The aerial robot is modelled after a quadcopter (40 cm diameter), which has no upper altitude limit.

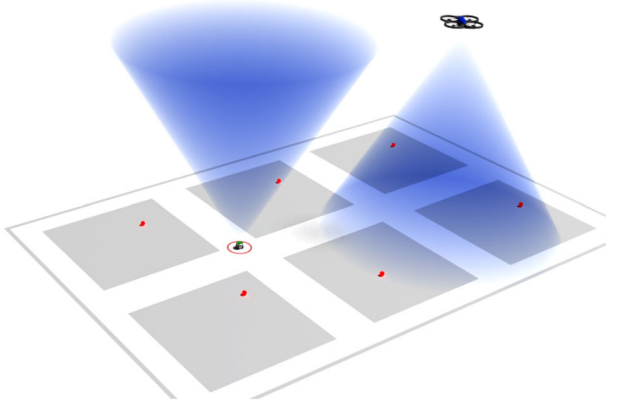

Fig. 1 Cooperative item collection task. The *red spheres* are the items to be collected. One item is placed in each of the *grey zones*. The *blue cones* depict the viewing range of the robots. The *red circle* around the ground robot depicts the range of its item sensor

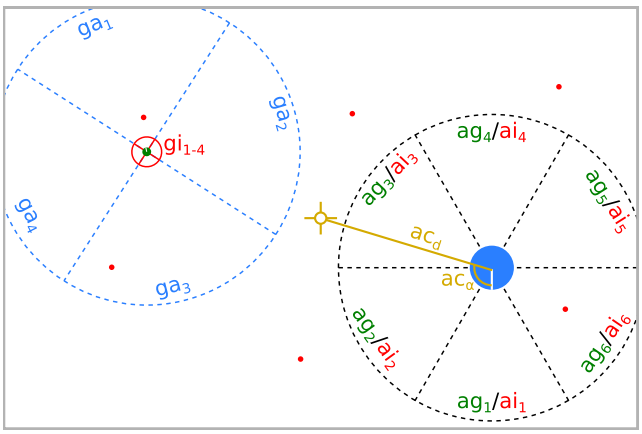

Fig. 2 Illustration of the robots' sensors. See Table 1 for the sensors description. Note that the range of the inputs ***ga***, ***ag***, and ***ai*** varies depending on the aerial robot's current altitude

3.2 Task variants

We use a number of task variants in which different skills must be learnt by the aerial robot before it can assist the ground robot in solving the task. This approach allows us to gain insight on how differences in the learning speed of the agents, and in the difficulty of establishing productive cooperation, affect the performance of cooperative coevolution.

In all task variants, six items are spread over an area of $550 \times 350\,\text{cm}^2$, with each item placed randomly in an area of $150 \times 150\,\text{cm}^2$, see Fig. 1. The environment is unbounded, and the robots are thus free to roam away from the items and from each other. A simulation ends when all items are collected, or when 200 s have elapsed. Each candidate solution (pair of ground/aerial robot controllers) is evaluated in ten independent simulations. The ground robot starts in a random corner of the arena, facing a random direction, while the initial conditions of the aerial robot depend on the task variant. The task variants are described below:

Fix-Tog: The aerial robot has no control over its altitude, but remains at the ideal sensing altitude (250 cm) throughout the whole simulation. The aerial robot starts above the ground robot.

Fix-Sep: Similar to *Fix-Tog*, but the aerial robot starts in a random location inside the arena, facing a random direction. This means that most of the times, the aerial and ground robots will not be within sensing range of each other at the beginning of a simulation.

Table 1 Configuration of the sensory inputs and actuators of the ground robot and aerial robot. See Fig. 2 for an illustration

Ground robot	Aerial robot
Sensory inputs	
$\boldsymbol{gi}_{1..4}$: 4 binary inputs that indicate the presence of items in the respective sector, within a 10 cm range	$\boldsymbol{ai}_{1..6}$: 6 real-valued inputs that give the distance of the closest item in the respective sector[a,b]
$\boldsymbol{ga}_{1..4}$: 4 binary inputs that indicate the presence of the aerial robot in the respective sector[a]	$\boldsymbol{ag}_{1..6}$: 6 real-valued inputs that give the horizontal distance to the ground robot in the respective sector[a,b]
	ah: 1 input that gives the current altitude[c]
	$\boldsymbol{ac}_{d,\alpha}$: 2 inputs that give the distance and relative angle to the centre of the arena
Actuators (maximum speed in parentheses)	
Linear speed (15 cm/s)	Front-back thrust (1 m/s)
	Left-right thrust (1 m/s)
	Up-down thrust (1 m/s)[c]
Turning speed (180°/s)	Yaw rotation (90°/s)

[a] Based on the camera sensor. Range up to 144 cm, depending on the aerial robot's current altitude

[b] If no robot/object is present in the input's respective section, the maximum sensor value is returned

[c] Not used in the *Fix-Tog* and *Fix-Sep* task variants

Var-Tog: The aerial robot starts on the ground, next to the ground robot, and can freely move up and down.

Var-Mid: The aerial robot starts on the ground, but it is placed in a random location in the arena, up to a distance of 300 cm away from the ground robot.

Var-Sep: Similar to *Var-Mid*, but the aerial robot is placed in a random location in the arena, with no restrictions.

3.3 Evolutionary setup

All the evolutionary approaches considered in this study are implemented over the same coevolution architecture, shown in Algorithm 1. Two populations are used. One population evolves the controller of the ground robot, while the other population evolves the controller for the aerial robot. Every generation, each population is evaluated in turn. To evaluate an individual from one population, a team is formed with one representative from the other population—the individual that obtained the highest fitness score in the previous generation, or a random one in the first generation. Only the individual currently being evaluated receives the fitness score obtained by the team. The choice of the single *previous best* individual as the representative for each population is the most common approach in robotics studies (Nitschke et al. 2012b; Potter et al. 2001; Yong and Miikkulainen 2009), as it only requires one evaluation per individual, and it was found to yield a good performance in preliminary experiments.[1] Recent results in a similar domain (Gomes et al. 2016b) have shown that the use of more than one representative per population does not improve coevolution's performance.

Algorithm 1 Base CCEA algorithm.

```
Let P be the set of populations.
for p ∈ P do
    p ← Generate initial population.
    r_p ← Randomly pick one individual i ∈ p
for each generation do
    for p ∈ P do
        for each individual i ∈ p do
            t ← Form a team with i and the representative
            r_q from each other population q ≠ p.
            e_i ← evaluate(t)
    for p ∈ P do
        r_p ← individual i ∈ p with maximum fitness score.
        p ← Breed p, based on the evaluation results e.
```

The neural network controllers of each population are evolved by NEAT (Stanley and Miikkulainen 2002), a state-of-the-art neuroevolution algorithm that evolves both the weights and topology of the networks, and that has been extensively used in evolutionary robotics. The two coevolving populations use the same NEAT parameters (Table 2), except for the number of input and output neurons. Each robot sensor is mapped to one input neuron of the network, and each output neuron is mapped to one robot actuator. This means that the ground robot's neural network has 8 inputs and 2 outputs, while the aerial robot's network has 15 inputs and 4 outputs (14 inputs and 3 outputs in the Fix-$*$ task variants), see Table 1 for details.

The fitness score of a team, F_i, corresponds to the number of items that were successfully collected during the simulation trial. During the evolutionary process, each candidate solution (robot team) is evaluated in ten independent simulations, and the fitness is given by the average score. The maximum fitness score that can be obtained is 6.0 (six collected items in all the trials), for all task variants. To obtain a more accurate estimate of the teams' quality, all the *best-of-generation* teams (the teams that obtained the highest fitness score in each generation, in each evolutionary run) were re-evaluated a-posteriori in 50 simulation trials. All the fitness plots presented in this paper correspond to the results obtained in these post-evaluations.

During the evaluation, the behaviour of each team was also measured in order to perform a-posteriori behavioural analysis. The behaviour characterisation does not influence the evolutionary process in any way, except when novelty search is used (Sect. 6). The team behaviour characterisation (Gomes et al. 2014b) is a vector of four real values normalised to [0,1]: (1) number of items collected; (2) time robots spent within the sensing range of one another; (3) average distance of one robot to the other; and (4) average distance of each robot to the closest item.

The task simulation was implemented using MASON[2], the evolutionary processes were implemented over the ECJ framework[3], and the implementation of NEAT was based on NEAT4J[4]. The source code of the experiments and the scripts used for analysis and plot generation are publicly available at https://github.com/jorgemcgomes/mase/releases/tag/NaturalComputing..

[1] We compared the *previous best* strategy with choosing a random population individual as representative (Wiegand et al. 2001). The *previous-best* strategy outperformed the *random* strategy across all task setups. The results are available at http://dx.doi.org/10.5281/zenodo.47066.

[2] http://cs.gmu.edu/∼eclab/projects/mason/.

[3] https://cs.gmu.edu/∼eclab/projects/ecj/.

[4] http://neat4j.sourceforge.net/.

Table 2 Parameters of the NEAT algorithm

Parameter	Value	Parameter	Value
Population size	150	Target num. species	5
Recurrency allowed	yes	Mutation prob.	25 %
Prob. add node	3 %	Prob. mutate bias	30 %
Prob. add link	5 %	Crossover prob.	20 %
Generations	700		

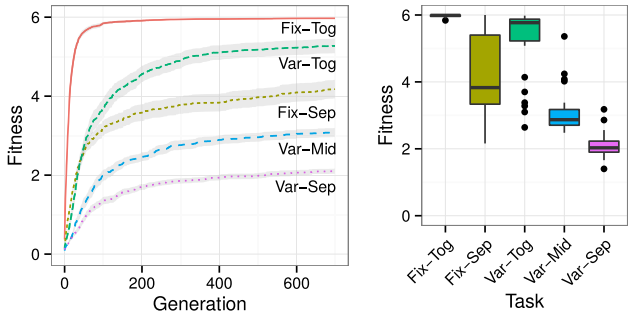

Fig. 3 Fitness scores achieved by the basic *CCEA* in each of the task variants. *Left*: average of the highest fitness scores achieved at each generation. The *grey* areas depict the standard error. *Right*: boxplots of the highest scores achieved in each evolutionary run

4 Fitness-driven CCEA

We begin by studying the performance of the basic CCEA in the different task variants. The highest fitness scores achieved throughout evolution are depicted in Fig. 3. Each evolutionary treatment was repeated in 30 independent evolutionary runs.

The results show that there are clear performance differences in the five variants, with statistically significant differences between all setups (Mann–Whitney U test, $p < 0.001$) in the highest fitness scores achieved. The CCEA can consistently and quickly evolve high-fitness solutions for the *Fix-Tog* variant. In the other task variants, where the aerial robot needs to learn more complex skills before being able to cooperate, the CCEA's performance was significantly affected. Coevolution displayed the lowest performance in the *Var-Sep* variant, where cooperation is harder to achieve.

To determine the reasons behind evolutionary failure in the more challenging setups, we divided the evolutionary runs into two sets: the successful runs, which achieved a fitness score of at least 4; and the failed runs, which did not reach that mark. We then visually inspected some of the highest scoring solutions evolved in each of these runs, see Fig. 4. These results reveal that the successful runs always relied on a high degree of cooperation to solve the task: the aerial robot was close to the ground robot most of the time. As it can be seen in the figure, the aerial robot first finds the ground robot (see the *Fix-Sep* and *Var-Mid*, successful runs) and then gets close to the items one at a time, while the ground robot is following it.

In the failed runs, however, we see a different scenario: the aerial robot displays a behaviour that almost seems to ignore the ground robot. It does not actively search for the items in the environment, but instead moves in circles over the arena. The ground robot uses the aerial robot's position to know where the arena is, but it has to search for the items alone using its very limited capabilities. The interaction between the two robots is therefore minimal or non-existent. Below, we investigate why a large number of evolutionary runs failed to evolve cooperation, thus resulting in poor solutions. We investigate two possible causes for the poor effectiveness of the CCEA: loss of fitness gradients, and premature convergence to mediocre stable states.

4.1 Loss of fitness gradients

Having agents that need to learn substantially different sets of skills can result in loss of fitness gradient in the populations (Wiegand 2003). If one agent does not possess the skills necessary to make any impact in the performance of the team, fitness diversity might be lost. Consequently, the individuals cannot be adequately ranked, and evolution starts to drift. In our item collection task, for instance, if the ground robot completely ignores the aerial robot, the behaviour of the aerial robot becomes inconsequential, and the aerial robot's population therefore loses fitness diversity. We investigated the loss of gradients by measuring the relative standard deviation (RSD) of fitness scores in each population, at every generation:

$$RSD(p,g) = \frac{\sigma(fit_{p,g})}{\mu(fit_{p,g})}, \quad (1)$$

where $fit_{p,g}$ is the set of fitness scores obtained by the individuals of population p at generation g, σ is the standard deviation, and μ is the arithmetic mean. A value of RSD close to zero means that fitness diversity is almost absent, suggesting loss of fitness gradients, while higher values indicate a rich fitness diversity.

The results in Fig. 5 show that the dispersion of fitness scores (RSD) is generally similar or higher in the failed runs than in the successful runs, for both populations, suggesting that the loss of fitness gradients is *not* the main cause for evolutionary failure.

4.2 Convergence to mediocre stable states

The other reason we investigate for evolutionary failure is convergence to mediocre stable states (Panait 2010). If the

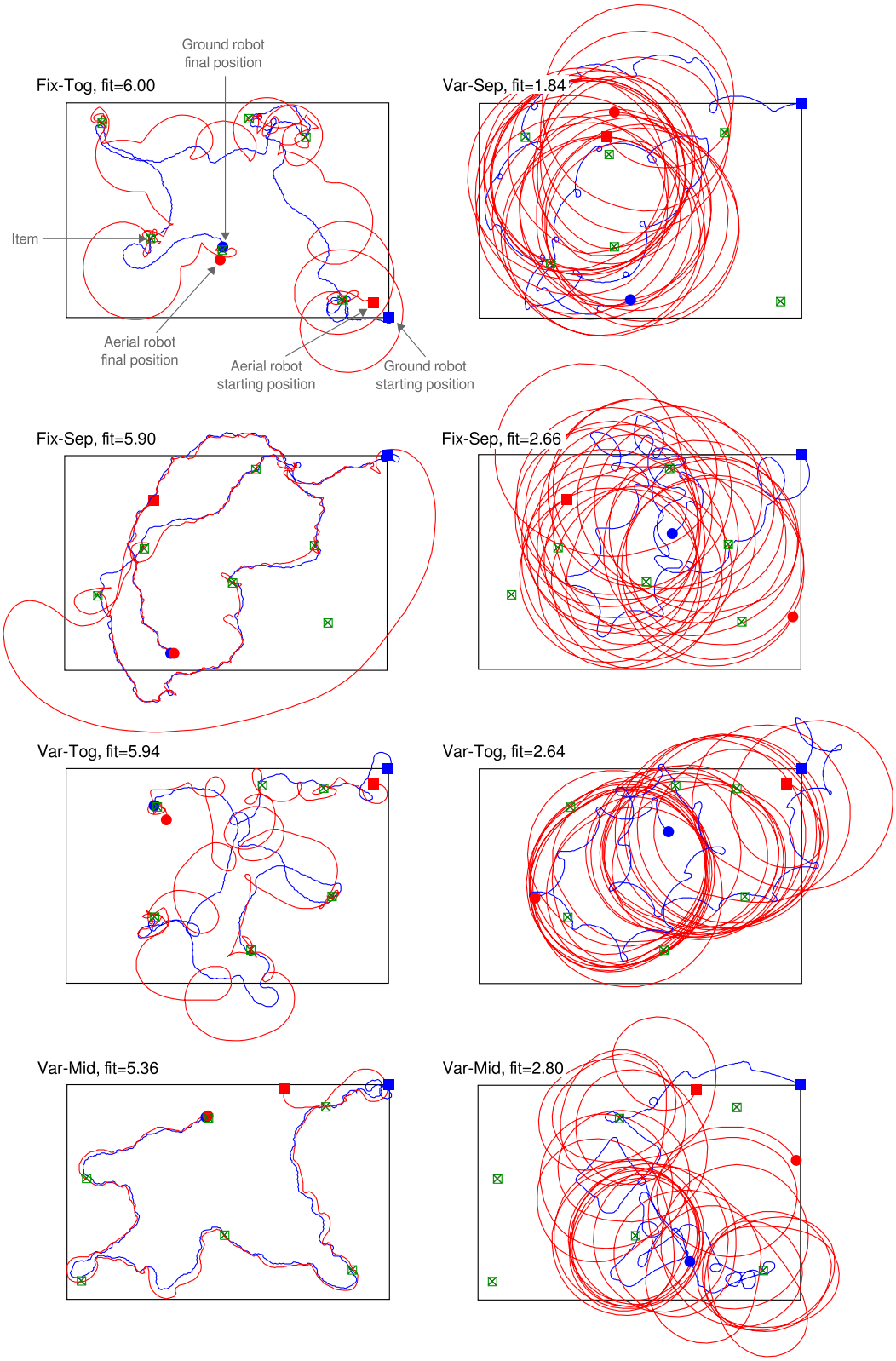

Fig. 4 Examples of the highest-scoring solutions evolved in evolutionary runs of fitness-driven coevolution. *Left column*: best solutions evolved in successful runs. *Right column*: best solutions evolved in failed runs. The *red line* depicts the aerial robot, and the *blue line* the ground robot. The *filled squares mark* the initial positions, and the *circles mark* the final positions. The *green squares* with crosses mark the items' positions. Videos are available at http://dx.doi.org/10.5281/zenodo.47066

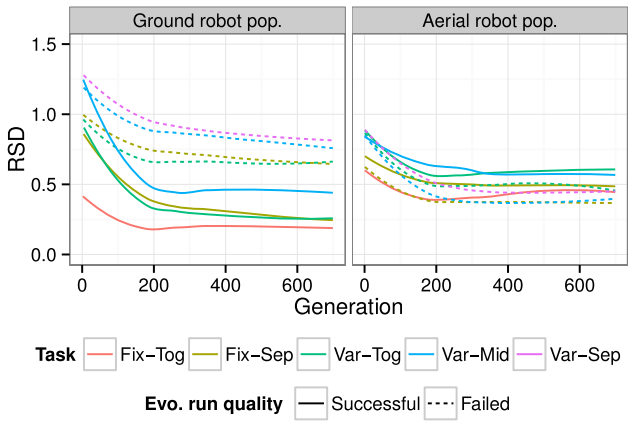

Fig. 5 Average relative standard deviation (RSD) of fitness scores inside each population, at every generation

two populations fail to sustain a mutual development of skills, the individuals of one population can become over-adapted to the poor behaviours found in the other population, reaching an equilibrium from which it can be hard to escape. In our item collection task, for instance, the flying robot can evolve a behaviour where it simply does circles inside the arena. The ground robot adapts to this behaviour, thus reaching an equilibrium state—a local optima in the collaboration space (see Fig. 4, failed runs).

To analyse if convergence to stable states was the culprit of the CCEA's low performance, we resorted to the analysis of the *best-of-generation* teams (the team that achieved the highest fitness score in each generation, in each evolutionary run), as done in previous works (Gomes et al. 2016b; Popovici and Jong 2006). For each of the *best-of-generation* teams, we measured the fitness (number of items collected) and the amount of time the robots spent within the sensing range of one another (*time within range*). This measure is directly related to the degree of cooperation between the robots, since the aerial robot cannot assist the ground robot if it is permanently outside its sensing range. The *average behaviour* of each generation was then calculated based on the mean values of fitness and *time within range* obtained in all *best-of-generation* teams (from the different evolutionary runs) of that generation. By plotting the average behaviour obtained throughout the generations of the evolutionary algorithm, it is possible to see to which types of solutions the evolutionary process is converging. The results are shown in Fig. 6.

The results from the successful runs show that, in all task setups, there is a close relation between the amount of cooperation (*time within range*) and the fitness of the solutions. High-scoring solutions always display high levels of cooperation. In the *Fix-Tog* task, evolution quickly converges to (near-)optimal solutions without much exploration. In this task, there are high levels of cooperation right from the beginning, as the robots start near one another and the aerial robot has the optimal sensing altitude. In the other task variants, evolution takes significantly more generations to evolve solutions with high levels of cooperation.

In the failed runs, evolution does not reach solutions with high cooperation levels, which is consistent with the behaviours observed in Fig. 4. The failed runs appear to be more biased towards the collection of items than robot cooperation, as evidenced by the higher numbers of items collected for the same levels of *time within range*. In the *Fix-Sep* and *Var-Tog* tasks, for instance, we can see that in the failed runs, evolution is trying to increase the number of items collected without increasing cooperation. In this item collection task it is, however, impossible to achieve high fitness scores without cooperation, and the results therefore indicate that evolution is trapped in a stable state from which it cannot escape. Overall, these results suggest that the main cause of evolutionary failure was convergence to mediocre stable states where there is no productive cooperation between the robots.

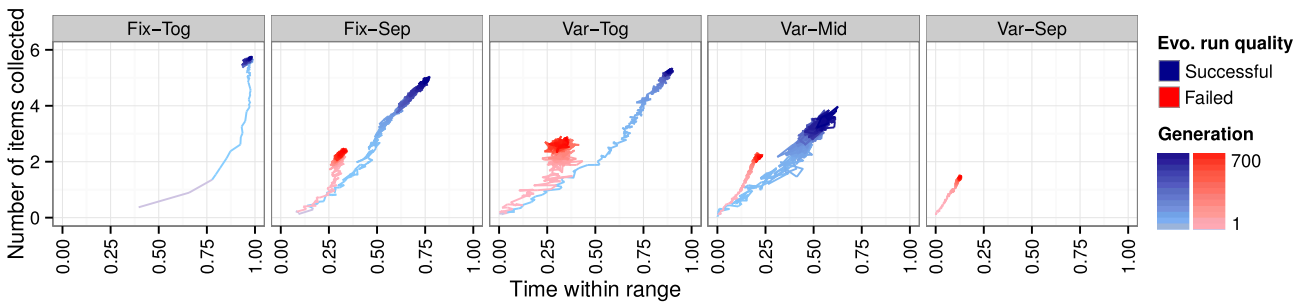

Fig. 6 Average behaviour of the *best-of-generation* solutions, grouped by successful (*blue*, highest fitness achieved ≥ 4) and failed runs (*red*, fitness <4). The *lighter colours* denote the earlier generations. The *time within range* is the time the robots spent within the sensing range of each other. Plots for individual evolutionary runs are available at http://dx.doi.org/10.5281/zenodo.47066

Table 3 Parameters of the genetic algorithm

Parameter	Value	Parameter	Value
Population size	150	Elite size	5
Tournament size	5	Gene mutation pr.	5 %
Mutation type	Gaussian	Crossover pr.	50 %
Crossover type	One point		
Hidden neurons (gr.)	5	Hidden neurons (ae.)	8
Genome size (gr.)	61	Genome size (ae.)	180

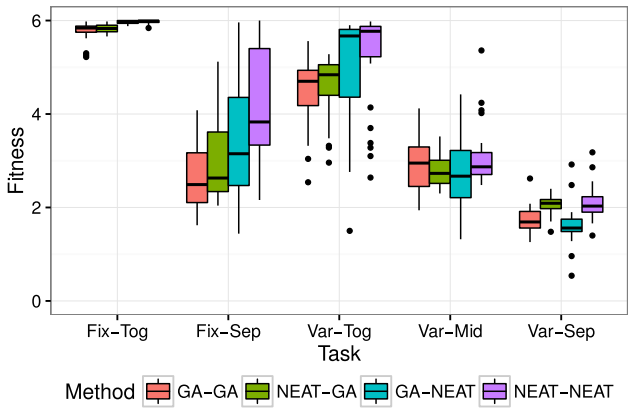

Fig. 7 Highest fitness scores achieved with the different combinations of evolutionary algorithms. GA–GA and NEAT–NEAT: same EA in the two populations; GA–NEAT: GA in the ground robot population and NEAT in the aerial robot; NEAT–GA: the opposite combination

5 Heterogeneity at the algorithmic level

One of the potential advantages of coevolutionary algorithms is that different populations can evolve with different individual representations, and even with completely different evolutionary algorithms (Vanneschi et al. 2006). In the experiments presented in the previous section, the two populations used different individual representations: the neural networks of the ground and aerial robots had a different number of input and output neurons. We did, however, use the same evolutionary algorithm (NEAT) in the two populations. In this section, we explore the effect of using different evolutionary algorithms and different evolutionary parameters in the populations.

5.1 Different neuroevolution algorithms

First, we try using different evolutionary algorithms in the different populations. Different EAs tend to explore the search space in different ways, which can potentially hinder synchronised learning (Uchibe et al. 1998). Here, we experiment with using a simple fixed-encoding genetic algorithm (GA) in one population, and the NEAT algorithm in the other, which has been shown able to evolve solutions in much fewer generations than simple genetic algorithms (Stanley and Miikkulainen 2002). In the GA, the neural networks have a fixed topology, using recurrency in the output neurons (Jordan 1997), see Table 3 for parameters. The following setups are evaluated: (1) GA–GA and (2) NEAT–NEAT, where the same algorithm is used in both populations; (3) NEAT–GA, where the ground robot population evolves with NEAT and the aerial robot evolves with the GA; and (4) GA–NEAT, where the opposite combination is used.

In Fig. 7, we show the highest fitness scores achieved by the evolutionary runs of each setup, for each task variant. The results first reveal that the NEAT–NEAT combination always yields a better performance than the GA–GA combination ($p < 0.001$, Mann–Whitney U test), except in the *Var-Mid* task ($p = 0.49$). This is consistent with previous works (Stanley and Miikkulainen 2002) that have shown NEAT's superior ability to evolve capable neural networks. The setups that combine GA and NEAT never yield a performance that is significantly inferior to the GA–GA setup. In fact, using NEAT in only one of the populations significantly improved the performance of the CCEA in some cases, when compared to the GA–GA setup (see *Fix-Sep* and *Var-Tog* tasks for instance).

5.2 Different population sizes

In this section, we study how the same evolutionary algorithm (NEAT) can be tuned individually for each population, in order to suit the algorithm to the characteristics of the agent that is being evolved. In this item collection task, for instance, the aerial robot has a significantly more complex sensor-actuator configuration than the ground robot. We study how the population size can be tuned to suit the controller's complexity. By reducing the

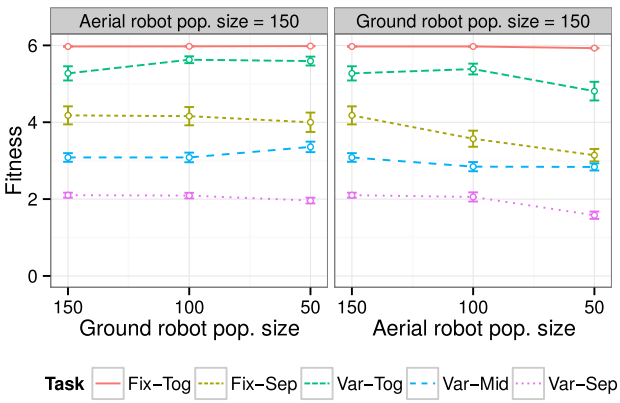

Fig. 8 Highest fitness scores achieved for each combination of population sizes. The NEAT algorithm is used in the two populations

population size of the least complex agent, the number of evaluations needed to achieve successful team solutions might be reduced.

In Fig. 8, we show the influence of the population size by varying the size of one population at a time, while keeping the other population at the maximum size (150). The NEAT algorithm was always used in the two populations. In all tasks, decreasing the size of the ground robot population never had a statistically significant impact in coevolution's performance (Mann–Whitney U test, $p > 0.2$), regarding the fitness scores achieved. Decreasing the size of the aerial robot population, on the other hand, had a significant impact in all task variants ($p < 0.05$). Overall, these results suggest a correlation between the complexity of the behaviour that an agent must learn and the optimal size of the respective population.

6 Improving cooperative coevolution

In this section, we experiment with improving the effectiveness of cooperative coevolution by avoiding premature convergence to mediocre stable states. We evaluate a number of methods that have been proposed in previous works as a way to avoid premature convergence in evolutionary robotics (Doncieux and Mouret 2014): incremental evolution (Gomez and Miikkulainen 1997), multi-objective optimisation of sub-goals (Mouret and Doncieux 2008), and novelty-driven cooperative coevolution (Gomes et al. 2014a, 2016b). The *Fix-Tog* task was not used in these experiments since the basic CCEA could always reach near-optimal solutions (Sect. 4).

6.1 Methods

6.1.1 Incremental evolution (Inc)

Incremental evolution (Gomez and Miikkulainen 1997) relies on the decomposition of the larger goal in sub-goals that are easier to achieve, thus overcoming bootstrap problems and premature convergence. We defined a sequence of sub-goals that try to bridge the gap between the number and complexity of skills each robot has to evolve. We encouraged the development of skills in the aerial robot, and the evolution of cooperation between the two robots, before trying to solve the ultimate objective of collecting items (F_i). We relied on the following sub-goals:

1. Minimise the difference between the aerial robot's altitude (a_t) and the near-maximum sensing altitude (A, 240 cm) over the simulation trial (T steps). The goal is achieved when 20 % of the individuals in a generation achieve a score of at least 0.9.

$$F_a = 1 - min\left(1, \sum_{t \in [1,T]} \frac{|a_t - A|}{T \cdot A}\right) \quad (2)$$

2. Maximise the time robots spend within the sensing range of one another (t_w). The goal is achieved when 20 % of the individuals in a generation achieve a score of at least 0.7.

$$F_w = t_w / T \quad (3)$$

3. Maximise the number of items collected throughout the simulation run (F_i).

The sequence of sub-goals depends on the task variant. For the *Fix-Sep* variant, we considered only two stages: $F_w \rightarrow F_i$, as the altitude of the aerial robot is fixed in this variant. For the other task variants, we used the three stages: $F_a \rightarrow F_w \rightarrow F_i$. The chosen representative individual of each population was the individual that obtained the highest fitness score in the previous generation, according to the fitness function of the current sub-goal.

6.1.2 Non-cooperative incremental evolution (NInc)

This approach is similar to incremental evolution described above, with one key difference: the ground robot only starts evolving when the final stage (F_i, collecting items) is reached. During the preceding stages, only the aerial robot evolves, while the ground robot remains still in its initial position. The rationale behind this approach is to develop essential skills in the aerial robot before starting the coevolutionary process.

6.1.3 Multi-objective evolutionary algorithm (MOEA)

In this approach, the three objectives that are used in incremental evolution, are employed in a multi-objective evolutionary algorithm, based on NSGA-II (Deb et al. 2002). The evolutionary process therefore tries to maximise the three objectives at the same time, instead of maximising them in a predefined sequence as in incremental evolution (Mouret and Doncieux 2008). For the *Fix-Sep* variant, only two objectives (F_w and F_i) were used, since the altitude is fixed. In order to implement the NSGA-II algorithm in NEAT, the individuals were scored according to their Pareto front and crowding distance, maintaining the same ranking order as in NSGA-II, and the selection process relied on these scores, as done in (Gomes et al. 2016b).

Regarding the choice of the representative individual, we evaluated four alternatives in preliminary experiments[5]:

[5] The results of the preliminary experiments are available at: http://dx.doi.org/10.5281/zenodo.47066

(1) choose the individual that achieved the highest fitness score (number of collected items) in the previous generation (as in Algorithm 1); (2) choose one random individual as the population representative; (3) randomly choose one individual from the non-dominated Pareto front; and (4) choose the N individuals that achieved the highest scores in each of the N objectives, in the previous generation. The best-performing strategy was the *previous best* strategy (1), which was thus adopted for our experiments.

6.1.4 Novelty-driven coevolution (NS)

Novelty-driven coevolution is implemented as proposed in (Gomes et al. 2016b), using the *NS-T* technique, which computes the individual's novelty scores based on the behavioural novelty displayed by the team in which the individual participated. The novelty score of each individual is combined with its fitness score with a multiobjective evolutionary algorithm, NSGA-II (Deb et al. 2002), as done in a number of previous works (Gomes et al. 2015c; Mouret 2011). The chosen representative individual of each population was the individual that obtained the highest fitness score (number of collected items) in the previous generation, as proposed in (Gomes et al. 2016b). The parameters of the novelty search algorithm were set according to (Gomes et al. 2015c). We use a value of $k = 15$ (nearest neighbours) for the novelty score computation, and the archive is randomly composed: every generation, four random individuals from each population are added to the archive. The archive size is capped at 2000 individuals, after reaching this limit random individuals are removed from the archive to make space for new ones.

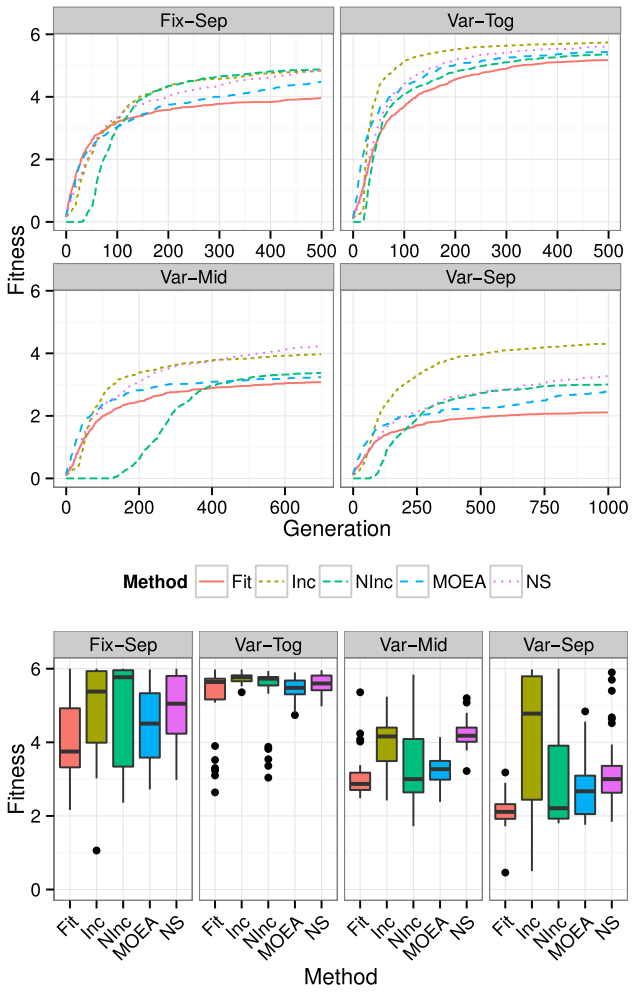

Fig. 9 *Top*: average of the highest fitness scores achieved at each generation, for each task variant and method. *Bottom*: highest fitness scores achieved in the evolutionary runs. Fitness corresponds to the number of items collected (F_i)

6.2 Results

We conducted 30 independent evolutionary runs in each experimental setup (methods × task variants). The fitness scores achieved with each method over the evolutionary process are shown in Fig. 9. Note that for all methods, the fitness score corresponds to the number of items collected (F_i), not necessarily the selection scores the individuals received during the evolutionary process. To understand how the studied methods can mitigate premature convergence and encourage cooperation, we performed an analysis of the *best-of-generation* solutions (similar to the one found in Sect. 4.2), see Fig. 10.

6.3 Analysis

6.3.1 Incremental evolution (Inc)

Incremental evolution was on average the highest performing approach, and outperformed the basic CCEA in all task variants (Mann–Whitney U test, $p < 0.05$). The results in Fig. 9 show that incremental evolution tends to reach high fitness scores in fewer generations than the other methods. Incremental evolution initially rewards the robots for staying within sensing range of one another. As the robots are essentially forced to cooperate before reaching the final stage, evolution is less likely to get stuck in a mediocre stable state where the robots do not cooperate when collecting the items.

As Fig. 10 shows, all evolutionary runs of *Inc* initially maximised the *time within range*, without increasing the number of items collected. When evolution reached the final stage, solutions started to maximise the number of items collected, and at this point two opposite scenarios could be observed: (1) evolution increased the fitness of solutions while increasing or maintaining the levels of *time within range*, leading to good solutions (successful runs); or (2) evolution increased the fitness of solutions but the

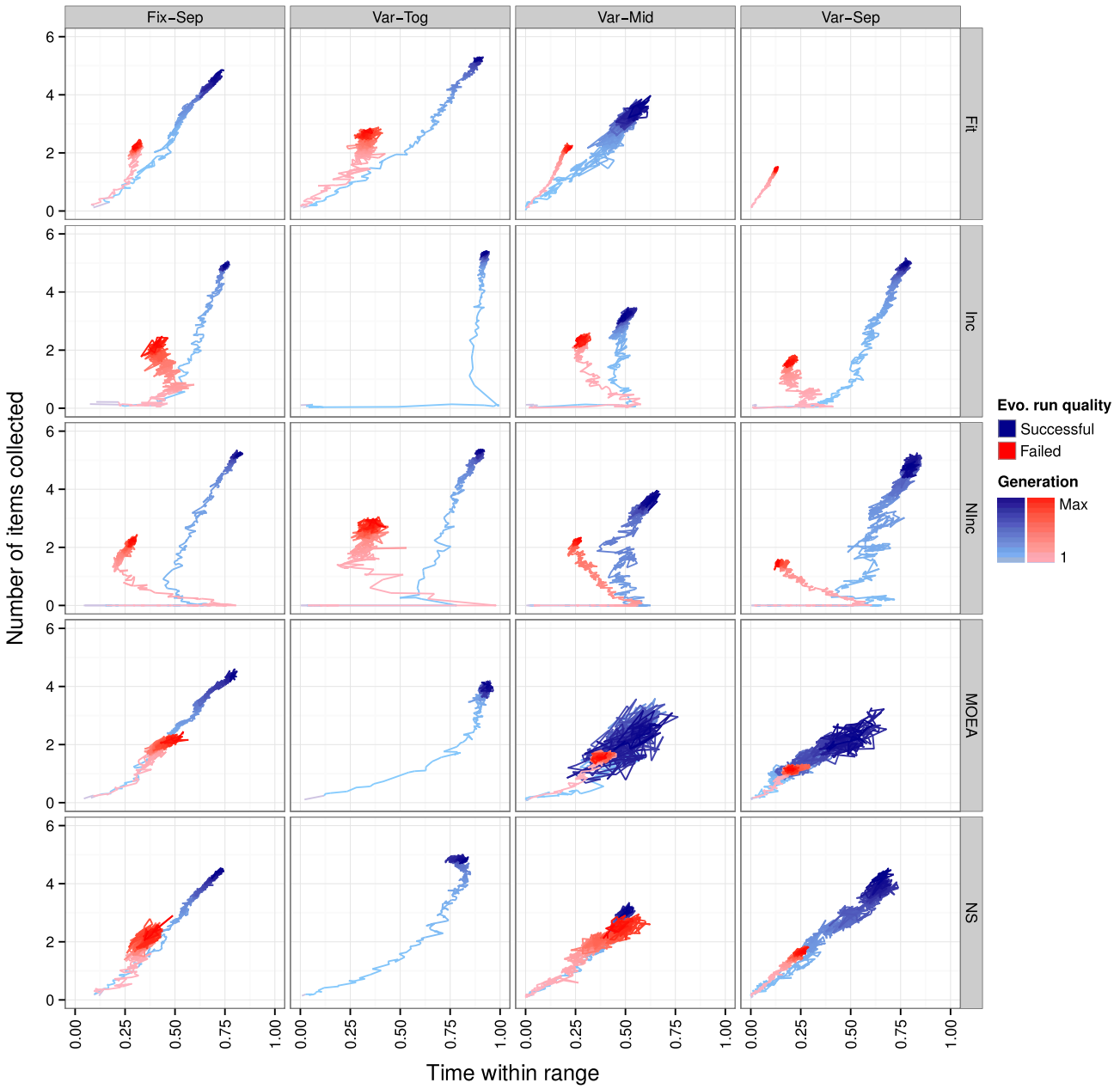

Fig. 10 Average behaviour of the *best-of-generation* solutions, grouped by successful (*blue*, highest fitness achieved ≥ 4) and failed runs (*red*, fitness <4). The *lighter colours* denote the earlier generations. The *time within range* is the time the robots spent within the sensing range of each other. Plots for individual evolutionary runs are available at http://dx.doi.org/10.5281/zenodo.47066

time within range decreased, ultimately leading to a mediocre stable state (failed runs).

These results show that, despite having reached the final stage in all evolutionary runs (see Fig. 11), incremental evolution often failed to achieve successful solutions, especially in the *Var-Sep* task (see Fig. 9, bottom). Even though the robots learn to find each other, they would not always evolve to successfully collect items. The effectiveness of incremental evolution depends, however, on the task decomposition defined by the experimenter, both in terms of the definition of the sub-goals, as well as the transitions between those goals. A more fine-grained incremental configuration (with more sub-goals for instance) could yield better performance, but it would also introduce additional biases in the evolutionary process.

6.3.2 *Non-cooperative incremental evolution (NInc)*

Non-cooperative incremental evolution achieved fitness scores similar to incremental evolution (*Inc*) in the *Fix-Sep*

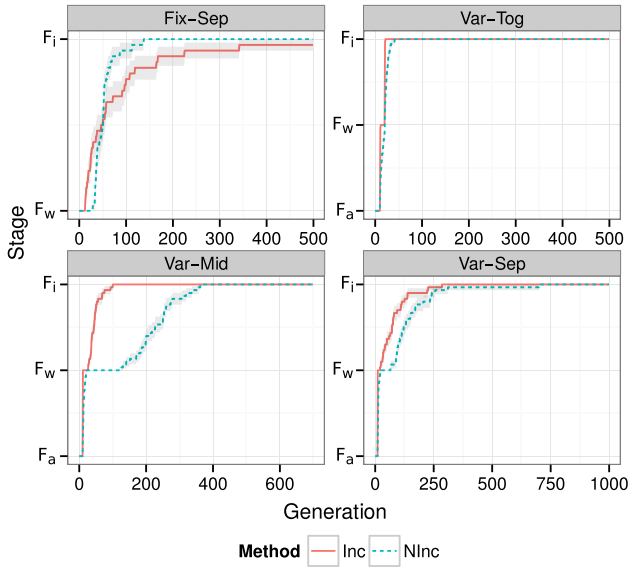

Fig. 11 Stage in which the evolutionary process was at each generation, averaged over the 30 evolutionary runs for each method/task. The *gray ribbon* shows the standard deviation. Note that in the *Fix-Sep* task there are only two stages

($p = 0.91$) and *Var-Tog* ($p = 0.09$) task variants, and reached significantly inferior fitness scores in the other two task variants ($p < 0.005$). The rationale for the *NInc* approach was to bootstrap the aerial robot before starting cooperative coevolution. As the results in Fig. 11 reveal, however, having a coevolving ground robot is preferable over having a static ground robot, when optimising the F_w objective (stay within sensing range of each other). *NInc* took on average 139 generations to fulfil the F_w objective, while *Inc* required only 66 generations (significant difference, $p < 0.001$).

With the *NInc* approach, the aerial robot evolves a behaviour that is over-adapted to a static ground robot. When cooperative coevolution starts, and the ground robot starts moving, the aerial robot is not prepared to cooperate with the ground robot, and its previously learned behaviours are thus prone to fail. Fig. 10 (*NInc* row) shows this degenerate behaviour: in all tasks, there is an initial increase in the *time within range*, and then a sudden decrease that coincides with the beginning of the coevolutionary process.

6.3.3 Multi-objectivisation (MOEA)

Despite the potential advantages of the multi-objectivisation approach over incremental evolution (Mouret and Doncieux 2008), our results failed to show such advantages. While MOEA achieved a similar performance to incremental evolution in the *Fix-Sep* task ($p = 0.10$), its performance was inferior in all the *Var-*$*$ tasks ($p < 0.001$). The MOEA approach also failed to improve over the basic fitness-driven coevolutionary algorithm in all tasks ($p > 0.05$) except *Var-Sep* ($p = 0.005$), see Fig. 9.

The results in Fig. 10 (MOEA row) reveal a relatively high variation of the average behaviour of the *best-of-generation* teams, especially in the *Var-Mid* and *Var-Sep* tasks, both in the number of items collected and the time within range. Although additional experiments would be needed to confirm the causes of MOEA's poor performance, these results suggest that the evolutionary algorithm was not able to strictly hold and advance the Pareto front across generations. This issue can potentially be explained by the fact that, unlike a traditional application of a MOEA, in a CCEA the objectives are not static: the objective scores given to any individual depend on the individuals with which it was evaluated.

6.3.4 Novelty-driven coevolution (NS)

Novelty-driven coevolution represents an alternative to avoid premature convergence that is substantially different from the approaches discussed above. Novelty search rewards the exploration of the behaviour space, without introducing biases towards specific behaviours (Gomes et al. 2016b). Regarding the fitness scores achieved, *NS-T* significantly outperformed the basic CCEA in all variants ($p < 0.01$), except in *Var-Tog* in which they achieved solutions of similar quality ($p = 0.54$). The performance of *NS-T* was only outperformed by *Inc* in the *Var-Sep* task ($p = 0.02$).

Another potential advantage of novelty-driven coevolution is the ability to discover a high diversity of solutions for a given problem (Gomes et al. 2016a, b). We confirmed this advantage by analysing the behavioural diversity evolved in each run, using a measure similar to the one proposed in (Gomes et al. 2016b). Considering the set φ of

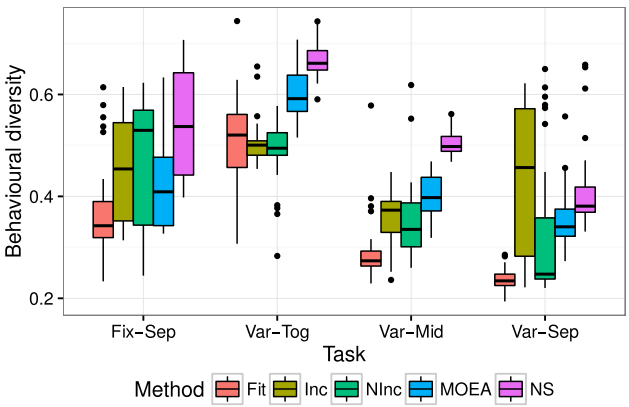

Fig. 12 Behavioural diversity, calculated based on the mean difference between all individuals evolved over the course of each evolutionary run

all individuals evolved in a given evolutionary run, behavioural diversity is given by the mean difference (D) between every two individuals:

$$D(\varphi) = \sum_{i \in \varphi} \sum_{j \in \varphi} \frac{dist(i,j)}{|\varphi|(|\varphi| - 1)} , \quad (4)$$

where *dist* is the Euclidean distance between the respective team behaviour characterisation vectors, presented in Sect. 3.3. The mean difference is a non-parametric measure of statistical dispersion, that is not defined in terms of a measure of central tendency. A low mean difference value indicates that most evolved individuals have very similar behaviours characterisations, while a high mean difference suggests that the evolutionary process was able to evolve a wide range of behaviourally different solutions. The results in Fig. 12 show that novelty-driven coevolution exhibited a significantly higher degree of behaviour exploration than all other approaches in all task variants ($p < 0.05$), except in the *Var-Sep* task where the diversity evolved by novelty-driven coevolution matched that of incremental evolution ($p = 0.7$). These results highlight novelty search's ability to explore the behaviour space.

As stated above, in the *Var-Sep* task variant, *NS-T* was not always able to achieve successful solutions and a high-diversity of behaviours. In this item collection task, the team behaviour space can only be adequately explored if the two robots cooperate. If, for instance, the aerial robot does nothing at all, or simply flies away, the diversity of team behaviours that can be achieved is significantly limited. In these situations, the exploration of the behaviour space becomes impaired, and novelty-driven coevolution can thus fail to discover high-quality solutions.

7 Discussion

7.1 Coevolution with heterogeneous populations

In the item collection task we use in this study, even the simplest task variant is associated with a significant heterogeneity in the robot team, with respect to sensor-effector capabilities, morphology, behaviours required for successful task execution, and complexity of neural network controllers. Nevertheless, cooperative coevolution consistently found (near-)optimal solutions for the simplest task variant. When the two robots can start cooperating from the very beginning of the evolutionary process, coevolution is able to sustain a mutual development of skills in the populations, leading them towards a (near-) optimal stable state.

We also showed that coevolution can cope with an additional kind of heterogeneity: the use of different evolutionary algorithms in the different populations. We experimented coevolution with the NEAT algorithm alongside a much less capable genetic algorithm (GA) evolving fixed topology networks. Our results suggested that the coevolutionary process is not significantly stifled by having different evolutionary algorithms, associated with significantly different learning speeds, operating simultaneously. Additionally, we studied how heterogeneity with respect to evolutionary algorithms can be exploited to suit the configuration of each population to the type of agent that needs to be evolved. Our results showed that decreasing the population size of the least complex agent did not have any impact in the quality of solutions achieved, meaning that fewer evaluations were needed to achieve the same team fitness levels. This population tuning can potentially be used to optimise the coevolutionary process with respect to resource usage.

7.1.1 Limitations of the CCEA algorithm

The experiments with the other more challenging task variants, however, revealed the limitations of the coevolution architecture when evolving heterogeneous agents. In the task variants where the aerial robot had to develop more complex skills before being able to cooperate, coevolution frequently failed. We investigated loss of fitness gradients and convergence to mediocre stable states as possible causes for this failure. Our results showed that premature convergence was the main culprit. If cooperation cannot be easily achieved in the beginning of the evolutionary process, behaviours where the robots try to achieve the goal without relying on cooperation may evolve. The evolution of such behaviours can lead to stable states from which evolution cannot easily escape. The attraction to such stable states was relatively weak in the easier task variants, as most runs were successful. In the more challenging task variants, the attraction was stronger, and the basic coevolutionary always converged to such stable states and failed to evolve successful solutions.

7.1.2 Attempts to overcome premature convergence

We tried to avoid convergence to mediocre stable states using a number of techniques proposed in previous works. Incremental evolution (Gomez and Miikkulainen 1997) was the most successful approach, significantly outperforming the basic CCEA in all tasks. Incremental evolution is, however, associated with well-known limitations (Doncieux and Mouret 2014), as a great deal of domain knowledge is required to design effective evolutionary stages. The need to manually define sub-goals, as well as appropriate transitions between these sub-goals, introduces strong biases in the evolutionary process (Doncieux and

Mouret 2014; Nelson et al. 2009), which counteracts the purpose of using evolutionary algorithms as black-box optimisers of agent controllers. Based on our intuition about the task, we defined three sub-goals, and although all stages were reached during evolution, many of the evolutionary runs still failed, especially in the more difficult task variants. This highlights the difficulty in properly configuring the stages and transitions in an incremental evolution scheme. We evaluated the multiobjectivisation of sub-goals as a way to overcome the difficulty in defining the order and transition of sub-goals (Mouret and Doncieux 2008), but the method generally failed in this task, and was unable to improve over incremental evolution and even the basic CCEA.

The non-cooperative version of incremental evolution performed significantly worse. Our results revealed that bootstrapping the more complex agent (the aerial robot in this task) before the coevolutionary process is not effective, as it does not take into consideration the influence the agents have on the behaviours of one another. The aerial robot frequently became over-fitted to a static ground robot, leading to a significant performance decrease when the coevolutionary process began. As argued in previous works (Dorigo et al. 2013), when developing cooperative systems, cooperation between the agents must be taken into account from the very beginning of the behavioural synthesis process.

Finally, we tried novelty-driven coevolution (Gomes et al. 2014a, 2016b), which rewarded individuals for displaying novel team behaviours. The performance of novelty search was similar to incremental evolution: it represented an improvement over the basic CCEA in all tasks, but still failed frequently in the most challenging task. One advantage of novelty search over incremental evolution is that it relies less on the experimenter's knowledge and potential biases, thus leaving evolution more free to explore diverse solutions (Doncieux and Mouret 2014). This advantage was confirmed in the comparison between the behavioural diversity generated by each algorithm: novelty search evolved a broader diversity of behaviours than any of the other algorithms tested. The effectiveness of novelty search is, however, dependent on the provided behaviour characterisation (Kistemaker and Whiteson 2011). Recent works have proposed the use of task-independent measures as a way to overcome this dependence (Doncieux and Mouret 2010; Gomes and Christensen 2013; Gomes et al. 2014b).

7.1.3 Towards effective coevolution with heterogeneous populations

Our experiments showed that incorporating domain knowledge into the process (incremental evolution) or adopting a more open-ended evolutionary approach (novelty search) can mitigate premature convergence issues. Nevertheless, even when using these techniques, many of the evolutionary runs still failed in the most demanding task variants. One of the main issues was that the coevolutionary process often started converging to solutions where the robots did not cooperate with one another. The first step to avoid this problem is to rely, if possible, on a task setup where the multiple agents can start cooperating right from the beginning of evolution. Another possibility is to design the fitness function in such a way that it is impossible to improve the fitness of the team without relying on cooperation. That is, the fitness function can be tailored (Nelson et al. 2009) to avoid the mediocre stable states where productive cooperation does not exist.

Although our experiments are based on a single task domain, the main challenge that is addressed in this paper is common to many multirobot tasks (Gomes et al. 2016b): how to foster synchronised learning, encourage cooperation in tasks where it is not readily available, and avoid premature convergence to non-cooperative solutions. In future work, we will extend our study to other tasks, including tasks with more than two robots.

8 Conclusion

Our work studies the challenges associated with coevolving behaviours for cooperative multirobot systems where there is a significant heterogeneity in the coevolving populations. Our experiments relied on a task where a highly capable aerial robot must assist a relatively simple ground robot in collecting items. We used multiple task variants in which we varied the number and complexity of skills that the aerial robot had to develop before being able to cooperate with the ground robot. Our work contrasts with the vast majority of previous works that have only used cooperative coevolutionary algorithms with very similar populations and agents. To the best of our knowledge, the work presented in this paper is the first to successfully demonstrate the evolution of controllers for a highly heterogeneous multirobot system.

Our results suggest that cooperative coevolution can work with an arbitrary level of heterogeneity in the populations, as long as the individuals from the different populations can establish a productive cooperation right from the beginning of the evolutionary process, thus leading to a mutual development of skills. We also showed that the CCEA architecture is able to simultaneously accommodate different evolutionary algorithms, with different learning speeds, and that the size of each population can be tuned according to the complexity of the respective agent. In task setups where one of the robots had to develop more

complex skills before being able to cooperate, the coevolutionary process failed frequently, converging to mediocre stable states where cooperation was nearly absent. One approach that showed promising results in mitigating this issue was novelty-driven coevolution (Gomes et al. 2016b), which is consistent with the large number of recent works that have demonstrated the importance of behavioural diversity in evolutionary robotics (Doncieux and Mouret 2014).

Overall, our experiments show that cooperative coevolution can be successfully applied to the evolution of morphologically heterogeneous teams. While issues related to premature convergence to mediocre stable states remain challenging, we have shown they can be alleviated through proper configuration of the task setup and the evolutionary algorithm.

Acknowledgments This research was supported by Fundação para a Ciência e Tecnologia (FCT), under Grants SFRH/BD/89095/2012, UID/EEA/50008/2013, and UID/Multi/04046/2013.

References

Blumenthal HJ, Parker GB (2004) Co-evolving team capture strategies for dissimilar robots, vol 2. In: AAAI artificial multiagent learning symposium. AAAI Press

Candea C, Hu H, Iocchi L, Nardi D, Piaggio M (2001) Coordination in multi-agent RoboCup teams. Robot Auton Syst 36(2):67–86

Christensen AL, Dorigo M (2006) Incremental evolution of robot controllers for a highly integrated task. In: Nolfi S, Baldassarre G (eds) From animals to animats 9. Springer, Berlin, pp 473–484

Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Doncieux S, Mouret JB (2010) Behavioral diversity measures for evolutionary robotics. In: Congress on evolutionary computation (CEC). IEEE Press, pp 1–8

Doncieux S, Mouret JB (2014) Beyond black-box optimization: a review of selective pressures for evolutionary robotics. Evolu Intell 7(2):71–93

Dorigo M, Trianni V, Şahin E, Groß R, Labella TH, Baldassarre G, Nolfi S, Deneubourg JL, Mondada F, Floreano D et al (2004) Evolving self-organizing behaviors for a swarm-bot. Auton Robot 17(2–3):223–245

Dorigo M, Floreano D, Gambardella L, Mondada F et al (2013) Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. IEEE Robot Autom Mag 20(4):60–71

Duan HB, Liu SQ (2010) Unmanned air/ground vehicles heterogeneous cooperative techniques: current status and prospects. Sci China Technol Sci 53(5):1349–1355

Ducatelle F, Di Caro G, Pinciroli C, Gambardella L (2011) Self-organized cooperation between robotic swarms. Swarm Intell 5(2):73–96

Gomes J, Christensen AL (2013) Generic behaviour similarity measures for evolutionary swarm robotics. In: Blum C, Alba E (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, New York, pp 199–206

Gomes J, Mariano P, Christensen AL (2014a) Avoiding convergence in cooperative coevolution with novelty search. In: Bazzan ALC, Huhns MN, Lomuscio A, Scerri P (eds) International conference on autonomous agents and multiagent systems (AAMAS). IFAAMAS, pp 1149–1156

Gomes J, Mariano P, Christensen AL (2014b) Systematic derivation of behaviour characterisations in evolutionary robotics. In: Sayama H, Rieffel J, Risi S, Doursat R, Lipson H (eds) International conference on the synthesis and simulation of living systems (ALife). MIT Press, Berlin, pp 202–209

Gomes J, Mariano P, Christensen AL (2015a) Cooperative coevolution of morphologically heterogeneous robots. In: Andrews P, Caves L (eds) European conference on artificial life. MIT Press, pp 312–319

Gomes J, Mariano P, Christensen AL (2015b) Cooperative coevolution of partially heterogeneous multiagent systems. In: Weiss G, Yolum P, Bordini RH, Elkind E (eds) International conference on autonomous agents and multiagent systems (AAMAS). IFAAMAS, pp 297–305

Gomes J, Mariano P, Christensen AL (2015c) Devising effective novelty search algorithms: a comprehensive empirical study. In: Silva S, Esparcia-Alcázar AI (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, pp 943–950

Gomes J, Duarte M, Mariano P, Christensen AL (2016a) Cooperative coevolution of control for a real multirobot system. In: Handl J, Hart E (eds) Parallel problem solving from nature – PPSN XIV. Springer, pp 591–601

Gomes J, Mariano P, Christensen AL (2016b) Novelty-driven cooperative coevolution. Evolutionary computation, **(in press)**

Gomez F, Miikkulainen R (1997) Incremental evolution of complex general behavior. Adapt. Behavior 5(3–4):317–342

Grabowski R, Navarro-Serment LE, Paredis CJ, Khosla PK (2000) Heterogeneous teams of modular robots for mapping and exploration. Auton Robots 8(3):293–308

Howard A, Parker LE, Sukhatme GS (2006) Experiments with a large heterogeneous mobile robot team: exploration, mapping, deployment and detection. Int J Robot Res 25(5–6):431–447

Hsieh MA, Cowley A, Keller JF, Chaimowicz L, Grocholsky B, Kumar V, Taylor CJ, Endo Y, Arkin RC, Jung B et al (2007) Adaptive teams of autonomous aerial and ground robots for situational awareness. J Field Robot 24(11–12):991–1014

Jones EG, Browning B, Dias MB, Argall B, Veloso M, Stentz A (2006) Dynamically formed heterogeneous robot teams performing tightly-coordinated tasks. In: Proceedings 2006 IEEE international conference on robotics and automation (ICRA). IEEE Press, pp 570–575

Jordan MI (1997) Serial order: a parallel distributed processing approach. In: Donahoe JW, Dorsel VP (eds) Neural-network models of cognition biobehavioral foundations, advances in psychology, vol 121. North-Holland, pp 471–495

Kistemaker S, Whiteson S (2011) Critical factors in the performance of novelty search. In: Krasnogor N, Lanzi PL (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, pp 965–972

Knudson M, Tumer K (2010) Coevolution of heterogeneous multi-robot teams. In: Pelikan M, Branke J (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, pp 127–134

Lacroix S, Le Besnerais G (2011) Issues in cooperative air/ground robotic systems. In: Kaneko M, Nakamura Y (eds) Robotics research, Springer tracts in advanced robotics, vol 66. Springer, Berlin, pp 421–432

Lehman J, Stanley KO (2011) Abandoning objectives: evolution through the search for novelty alone. Evol Comput 19(2):189–223

Mathews N, Christensen AL, O'Grady R, Dorigo M (2010) Cooperation in a heterogeneous robot swarm through spatially targeted communication. In: Dorigo M, Birattari M (eds) Swarm intelligence, LNCS, vol 6234. Springer, Berlin, pp 400–407

Mouret JB (2011) Novelty-based multiobjectivization. In: Doncieux S, Bredeche N, Mouret JB (eds) New horizons in evolutionary robotics, studies in computation intelligence, vol 341. Springer, Berlin, pp 139–154

Mouret JB, Doncieux S (2008) Incremental evolution of animats behaviors as a multi-objective optimization. In: Asada M, Hallam JCT, Meyer JA, Tani J (eds) From Animals to Animats 10. Springer, Berlin, pp 210–219

Mouret JB, Doncieux S (2012) Encouraging behavioral diversity in evolutionary robotics: an empirical study. Evol Comput 20(1): 91–133

Nelson AL, Barlow GJ, Doitsidis L (2009) Fitness functions in evolutionary robotics: a survey and analysis. Robot Auton Syst 57(4):345–370

Nitschke GS, Schut MC, Eiben AE (2009) Collective neuro-evolution for evolving specialized sensor resolutions in a multi-rover task. Evol Intell 3(1):13–29

Nitschke GS, Eiben AE, Schut MC (2012a) Evolving team behaviors with specialization. Genet Program Evol Mach 13(4):493–536

Nitschke GS, Schut MC, Eiben AE (2012b) Evolving behavioral specialization in robot teams to solve a collective construction task. Swarm Evol Comput 2:25–38

Nolfi S, Floreano D (2000) Evolutionary robotics. MIT Press, Cambridge

Panait L (2010) Theoretical convergence guarantees for cooperative coevolutionary algorithms. Evol Comput 18(4):581–615

Parker L, Kannan B, Tang F, Bailey M (2004) Tightly-coupled navigation assistance in heterogeneous multi-robot teams. In: IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE Press, pp 1016–1022

Popovici E, De Jong K (2006) The dynamics of the best individuals in co-evolution. Nat Comput 5(3):229–255

Potter MA, Jong KAD (2000) Cooperative coevolution: an architecture for evolving coadapted subcomponents. Evol Comput 8(1): 1–29

Potter MA, Meeden LA, Schultz AC (2001) Heterogeneity in the coevolved behaviors of mobile robots: the emergence of specialists. In: Nebel B (ed) International joint conference on artificial intelligence (IJCAI). Morgan Kaufmann, San Francisco, pp 1337–1343

Stanley K, Miikkulainen R (2002) Evolving neural networks through augmenting topologies. Evol Comput 10(2):99–127

Trianni V, López-Ibáñez M (2015) Advantages of task-specific multi-objective optimisation in evolutionary robotics. PLoS one 10(8):e0136406

Uchibe E, Asada M (2006) Incremental coevolution with competitive and cooperative tasks in a multirobot environment. Proc IEEE 94(7):1412–1424

Uchibe E, Nakamura M, Asada M (1998) Co-evolution for cooperative behavior acquisition in a multiple mobile robot environment. In: IEEE/RSJ international conference on intelligent robots and systems (IROS). IEEE Press, pp 425–430

Vanneschi L, Mauri G, Valsecchi A, Cagnoni S (2006) Heterogeneous cooperative coevolution: strategies of integration between GP and GA. In: Keijzer M, Cattolico M (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, pp 361–368

Wiegand RP (2003) An analysis of cooperative coevolutionary algorithms. PhD thesis, George Mason University

Wiegand RP, Liles WC, De Jong KA (2001) An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. In: Spector L, Goodman ED (eds) Genetic and evolutionary computation conference (GECCO). ACM Press, pp 1235–1245

Yang J, Liu Y, Wu Z, Yao M (2012) The evolution of cooperative behaviours in physically heterogeneous multi-robot systems. Int J Adv Robot Syst 9(253):1–10

Yong CH, Miikkulainen R (2009) Coevolution of role-based cooperation in multiagent systems. IEEE Trans Auton Mental Dev 1(3):170–186