The Evolution of Cooperative Behaviours in Physically Heterogeneous Multi-robot Systems

Abstract Evolutionary Robotics (ER) is a methodology that uses evolutionary computation to develop controllers for autonomous robots. The existing works focus on single robot systems or physically homogenous multi-robot teams, although physical heterogeneity is more prevalent in the real world. It is therefore instructive to examine whether cooperative behaviours can be synthesized using artificial evolution for a team of physically heterogeneous robots. This paper makes an important contribution in answering the question of whether robots with distinct capabilities can synthesize their control strategies to accommodate their own capabilities without human intervention. We present an empirical analysis of the collaboration mechanisms and suggest guidelines about how to choose appropriate evolution methods. Simulated experiments with a team of e-puck robots show that evolution can lead to effective controllers for robots with distinct capabilities.

Keywords Multi-robot System, Cooperative Behaviours, Evolutionary Robotics

1. Introduction

The unknown and dynamic characteristics of environments may mean that explicitly designing a control system for a single robot is difficult, let alone for multi-robot systems (MRS). Evolutionary robotics (ER) is a field of research that applies artificial evolution to generate control systems for autonomous robots [1, 2]. By adopting artificial evolution, it is possible to learn and synthesize the control systems of a group of robots that are able to display expected behaviours without human supervision. In addition, explaining the initiation and maintaining of cooperation in multi-robot systems will accelerate our understanding of biological systems and human societies [3, 4].

The use of artificial evolution in robotics has been continuously studied in the past [1, 2, 5-8]. For instance, a group of robots are evolved to display aggregation and coordinated movements [9-11], perform multi-objective tasks [12, 13], and learn to cooperate to play competitive games [14-17], etc. Evolution of unplanned coordination
among independent agents in market selection games is also examined [18]. Moreover, embodied evolution (EE), which is a new methodology for ER, constitutes a fully distributed evolutionary algorithm embodied in physical robots [19].

What has not been shown is that ER methods can be extended to generate robot controllers capable of complex autonomous behaviours; in particular, no ER work has yet shown that it is possible to evolve complex controllers in general cases or for generalized tasks [8]. Waibel et al. [20] distinguished three types of cooperative tasks and suggested guidelines for the optimal choice of genetic team composition and level of selection.

Although the above-mentioned works have their strengths and weaknesses, none of them fully satisfies our expected design requirements because they deal with a single robot system (e.g., developmental robotics [21, 22]) or physically homogenous multi-robot teams. In this paper, we focus on physically heterogeneous multi-robot teams that interact to solve cooperative multi-agent tasks. Here, “physically heterogeneous” refers to hardware (or capability, e.g., onboard sensors and actuators) diversity and “genetically heterogeneous” refers to genetics (i.e., control rules) diversity. Dynamically formed heterogeneous robot teams executing coordinated tasks may have little prior information about the tasks, other robots, and the environments in which they will operate.

Cooperative co-evolutionary algorithms (CCAs) [23-26] are applied by manually decomposing problem representations into subcomponents and then creating separate populations of individuals for each subcomponent. This allows a team to be composed of specialized sub-groups and corresponds to biological co-evolution of multiple species. Many characteristics differentiate the approach proposed in this paper from CCAs. While CCAs require user-specified decomposition of the problem, we do not rely on this decomposition. Moreover, these CCAs evolve the controller of physically homogenous robots (or software agents) under different sub-groups with different fitness evaluation functions; we evolve the controller of physically heterogeneous robots under different sub-groups sharing the same fitness evaluation functions.

Unlike the existing works in this field, our work seeks to answer the question of whether robots with distinct capabilities can synthesize their control strategy to accommodate their own capabilities without human intervention. To attain this objective, we have designed an evolution framework to synthesize cooperative behaviours for physically heterogeneous MRS and introduce the important factors that can influence the efficiency of evolution. We investigate this phenomenon in an attempt to understand these enabling features. Several options for the key steps are presented to evaluate the synthesis process of controllers using artificial evolution under different experimental conditions. Accordingly, we compare the performance and suggest guidelines about how to choose appropriate evolution methods.

The rest of this paper is structured as follows. In the next section, we introduce the design framework of evolution and present a detailed discussion on the implementation of the proposed method. The experimental set-up in Section 3 includes the description of simulated environment and experimental conditions. In Section 4 we discuss the experimental results and evaluate the performance of synthesized controllers, and conclude with a summary and outlook in Section 5.

2. Evolution of Cooperation

While evolving control strategies for MRS consisting of multiple robots with distinct capabilities, it is expected that the team can display cooperative behaviours, and that each type of robot can synthesize its control strategy to accommodate its own capabilities. Firstly, we focus our attention on design framework in Algorithm 1, and introduce the important factors that can influence the efficiency of evolution.

Algorithm 1. The design framework of evolution.

1. Choose which activation values of sensors are assigned to the controller as inputs.
2. Decide the controller’s outputs, which may reflect the behaviours to search and execute the task and/or to show cooperative actions.
3. Construct the high-level controller’s structure, and determine the approach to the genetic representation of controller.
4. Evolve the controllers for different types of robots using evolutionary algorithms.
   a. Create subpopulations for every type of robot.
   b. Formulate the fitness functions for subpopulations employed during evolution.
   c. Set the methods used for choosing representatives in subpopulations.
   d. Set the general parameters for the evolutionary algorithm.
   e. Co-evolve subpopulations in a round-robin fashion and evaluate each combined group by interaction with, and feedback from, a dynamic environment in simulation to drive synthesis of controllers.
5. Verify the performance of final synthesized controllers on simulated and/or real robots.

There are several different implementations of the steps in Algorithm 1. In order to verify the effectiveness, the proposed framework is applied to evolve desired
behaviours for physically heterogeneous multi-robot systems. We present different options for the key steps, evaluate the synthesis process of controllers using artificial evolution under different experimental conditions, and suggest guidelines about how to choose appropriate evolution methods.

2.1 Neural Controller

Neural networks have been proved in many studies to be efficient controllers of robots. The neural network may have hidden neurons [20], the connections and neuron units can be added or removed during evolution [15], and modular network architecture can be used to reduce interference in the adaptation of different parts of the controller system [27]. However, a simple feed-forward neural network with two layers is competent in our scenario, where only the weight connections of the neural controller are encoded and evolved. The neural network architecture can be chosen according to the actual situations.

The e-puck robots are small and two-wheeled, and equipped with eight infrared sensors to detect walls and other robots up to a distance of 60 mm, one colour camera, three directional microphones and a virtual gripper. Each robot is provided with two motors that control the speed of the two responding wheels and a loudspeaker that emits a sound (the loudspeaker has only two statuses: on or off) (see Figure 1). The e-puck robot can be configured to different types of robot with distinct capabilities.

![Figure 1](image1.jpg)

**Figure 1.** Left: The e-puck robot facing right. Right: A schema of the position of sensors and motors within the robot body. Eight full rectangles and three empty circles represent the eight infrared sensors and the directional microphones. The two long rectangles and the circle represent the two wheels controlled by the two separate motors and the speaker. The two dashed lines indicate the angle of view of the camera.

The high-level controller is implemented as a feed-forward neural network with 18 and 3 nodes in the input and output layers, respectively (see Figure 2). The weights of the neural network are evolved using evolutionary algorithms. The 18 sensory neurons encode the state of the eight proximity infrared sensors, three sound sensors, five vision sensors, a gripper and a bias unit (i.e., a unit whose activation state is always 1.0). Each sensory neuron is directly connected to two motor neurons and a speaker neuron. Thus, the neural controller is made up of 18×3=54 connects, each associated with a weight ranging in the interval \([-10,+10]\) and represented in the genotype with 8 bits. Therefore, the genotype is composed of 54×8=432 bits.

![Figure 2](image2.jpg)

**Figure 2.** The neural network controller.

2.2 Evolution

Each type of robot has its own subpopulation \(P^i (i=1, \ldots, N_{type})\), and robots of the same type share the same genes (i.e., they are genetically homogeneous). Each subpopulation is co-evolved in a round-robin fashion, that is, the fitness of a subpopulation member is evaluated by combining it with the representatives from the remaining (temporarily frozen) subpopulations (see Figure 3). There are many possible methods for choosing representatives with which to collaborate, including greedy and random choice strategies.

![Figure 3](image3.jpg)

**Figure 3.** Co-evolving subpopulations in a round-robin fashion.

Three typical methods are used for choosing representatives in subpopulations. The “static-select” method selects fixed partners as the representatives, that is, the \(j\)th individual in subpopulation \(P^j\) is evaluated by combining it with the \(j\)th individuals in the remaining subpopulations \(P^k\) (\(k=1, \ldots, N_{type}, k\neq i\)). However, the “best-select” method selects the best individuals in the remaining subpopulations \(P^k\) (\(k=1, \ldots, N_{type}, k\neq i\)) to be combined with the \(j\)th individual in subpopulation \(P^j\), which is being evaluated. The “Pr-select” method is a
more ecological approach, in which representatives are chosen non-deterministically based on their fitness. The probability for selecting the jth individual from subpopulation P is given by

$$\text{Pr}(i,j) = \frac{\exp(\lambda v_i)}{\sum_{j=1}^{N_{pop}} \exp(\lambda v_j)}$$

(1)

where $\lambda$ is a positive parameter called the inverse temperature, $v_i$ is the fitness of the jth individual from subpopulation $P$, and $N_{pop}$ is the subpopulation size. Low $\lambda$ causes (nearly) equi-probable selection of all individuals, while high $\lambda$ causes selection of the highest performance with probability close to one. We empirically set $\lambda=100$ at the generation t.

The combined group is allowed to live for four epochs, and each epoch consists of 120/0.05=2400 cycles (120 is the time each epoch lasts in seconds, and 0.05 is the length of the simulation step).

The evolutionary algorithm used to evolve the controllers utilizes a population of 100 randomly generated binary genotypes for every subpopulation. At every generation, the best 10% of genotypes will be directly copied to the new generation, and the best 30% of genotypes will be used to create new individuals, with a 20% probability of crossover. Each offspring is mutated with a 2% probability of flipping each bit. One evolutionary process lasts 400 generations, and five independent runs were performed starting with different randomly generated initial populations.

2.3 Performance Function Specification

We consider the different capabilities of robots in two facets: task discovering and task solving. The first facet focuses on the robots’ movement and sensor ranges, and the second concerns the state transition of tasks being executed. We abstract the tasks into three different states (i.e., S0, S1 and S2) inspired by fire extinction. The start state of a task is S0, and it will transition to the next state S1 if the task in state S0 has been continuously executed by any robot for $T_s$ seconds. Similarly, the transition from state S1 to S2 occurs once any robot continues executing the task in state S1 for $T_s$ seconds, and we say the task is completed when it is in state S2. However, the task’s state reverts from S1 to S0 after $T_r$ seconds if no robot continues to execute it (see Figure 4). We use colour change as the environment cue to simulate the state transition of tasks in experiments, facilitating robots to evolve their behaviours.

Robot teams consist of two types of e-puck robots, namely EP1 and EP2. The EP1 robots have greater speed and longer vision sensor ranges than EP2, but their task solving capabilities are quite poor compared to EP2: $T_g=2s$ and $T_o>=T_o$ (e.g., $T_e:=2s$, $T_o=50s$, $T_o=5s$ – see Figure 4). In order to make it easier for robots to evolve their controllers, we introduce a virtual gripper to every robot. When a robot is standing before a task setting within 10 mm, its gripper is closed; otherwise, it is open. Although this particular scenario involves only two types of robots, the actual number of types may be larger.

We take two different options to set the fitness function for subpopulations. The first “shared-fitness” option means that every subpopulation shares the same fitness function:

$$f_i = \alpha \times \Sigma_1 + \beta \times \Sigma_2$$

(2)

where $\Sigma_1$ is the total number of tasks found, $\Sigma_2$ is the total number of completed tasks, and $\alpha$ and $\beta$ are scalar weighting factors. The values of the factors are empirically set to 2 and 5, respectively.

A robot will be rewarded only if it finds a task for the first time, and all robots finding the same task will not be rewarded. This rule is set in case the abnormal behaviour of repeating to find the same task in order to gain reward is synthesized.

The second “independent-fitness” option means that two subpopulations have the separate fitness function:

$$f_i^1 = f_i + \gamma \times \Sigma_3 + \delta \times \Sigma_4$$

$$f_i^2 = f_i$$

(3)

(4)

where $f_i^1$ is the fitness function for EP1 robots, and $f_i^2$ is for EP2 robots. $\Sigma_3$ is the total cumulative number of times robots choose to emit sound when they are around a task, and $\Sigma_4$ is the total cumulative number of times that robots choose to emit sound when they are not around any task. $\gamma$ and $\delta$ are scalar weighting factors, which are empirically set to 2 and -2, respectively. $\gamma \times \Sigma_3$ indicates that a robot will be rewarded if it chooses to emit sound when it is around a task, and $\delta \times \Sigma_4$ indicates that the robot will be punished otherwise.

3. Experimental Setup

To perform experiments with a large team of robots, we evaluated our approach in realistic simulation experiments (see Figure 5) on an open evolutionary framework called Teem [28]. The robots can perceive the intensity of sound using three sound sensors that simulate three directional microphones using a set of equations [9], and the total perceived amplitude (TPA), i.e., the result of the contribution of different sound sources corresponding to different robots is computed.
according to the proposed function. To simulate the robot and the environment as accurately as possible, short range detection of obstacles or other robots is achieved by using eight proximity sensors, simulated using a sampling technique [1].

\[
\text{image} \left[ i \right]^* = \text{image} \left[ i \right] \times \frac{1}{1 + \text{fogDensity} \times \sqrt{\text{zbuffer}[i]}} \tag{5}
\]

where \( \text{image} \left[ i \right] \) is the original data, \( \text{image} \left[ i \right]^* \) is the computed data, \( \text{fogDensity} \) is the density of fog, and \( \text{zbuffer}[i] \) is the colour depth.

For example, in Figure 6, two robots face each other and each robot’s camera visualizer reflects other objects’ colour, position and distance in FOV. The grey value, spatial difference and/or average value of the whole image and/or section are assigned to the neural controller’s inputs.

We evaluate the synthesis process of cooperative behaviours using artificial evolution under different experimental conditions, as shown in Table 1. With the virtual gripper, a robot can easily estimate whether it is close to a task spot or not. Hence, the virtual gripper option is introduced to investigate the effect of this input information in the evolution of cooperative behaviours. With the speaker as an output, a robot’s controller can emit a sound to attract other robots. In contrast, when the option is off, no robot can use the sound to invite other robots to participate in solving the cooperative task, and the evolution of cooperative behaviours may come to be more difficult.

| Conditions          | Options      | Comments                                      |
|---------------------|-------------|-----------------------------------------------|
| Virtual gripper     | with-gripper| Use the gripper as input.                     |
|                     | without-gripper | Do not use the gripper as input.            |
| Representative      | static-select| A fixed partner is selected as the representative. |
| Selecting method    | best-select | The best one is selected as the representative. |
|                     | Pr-select   | The representative is selected non-deterministically based on the fitness. |
| Fitness function    | shared-fitness | Subpopulations share the same fitness function. |
|                     | independent-fitness | Subpopulations have different fitness function. |
| Speaker             | with-speaker | Use the speaker as output.                   |
|                     | without-speaker | Do not use the speaker as output. |

Table 1. Experimental conditions.

At the beginning of each epoch the robots and tasks are placed in randomly generated positions and orientations within the arena. Each task will be randomly placed in a new position immediately after it is completed. Hence, the total number of tasks in the environment remains unchanged. There are four tasks, five EP1 and five EP2 robots, and the size of environment is 200×200.

4. Results and Discussion

4.1 Experimental Results

As we have mentioned above, the robot team consists of two types of e-puck robots, namely EP1 and EP2. The EP1 robots have greater speed and longer vision sensor ranges than the EP2, but their task solving capabilities are quite poor compared to the EP2. Hence, we expect the robots’ control strategies to accommodate their own capabilities and maximum the global revenues; i.e., the EP1 robots should prefer to search for new tasks and emit sounds when they find them, and EP2 robots should be attracted by these sounds and follow them to help execute the new task. EP1 robots should be uninterested in following these sounds and instead choose to spread out to find other tasks.
We conducted Experiments 1-6 and the obtained results are presented in Figures 7-12. Table 2 lists all the comparative experiments that we have carried out. In order to remain concise, the default “with-speaker” option in Experiments 3-6 is not listed in the table. Note that the fitness plotted in Figures 7 and 8 is the fitness in a unit time; specifically, total fitness in an epoch is divided by the epoch cycles (i.e., 2400). For example, 0.03×2400/(2+7)=8 tasks are completed in an epoch if the corresponding fitness is 0.03.

| Experiment | Option                        |
|------------|-------------------------------|
| Exp 1      | without-gripper& static-select& shared-fitness & without-speaker |
| Exp 2      | without-gripper& static-select& shared-fitness & with-speaker     |
| Exp 3      | without-gripper& best-select& shared-fitness                      |
| Exp 4      | without-gripper&Pr-select& shared-fitness                         |
| Exp 5      | without-gripper& best-select& independent-fitness                 |
| Exp 6      | with-gripper& best-select& independent-fitness                    |

Table 2. List of comparative experiments.

The best performance of the robot team with and without speakers (i.e., in Exp 1 and Exp 2) did not differ significantly, as shown in Figure 7. The evolution of cooperative behaviours do not benefit from the “static-select” method. However, one interesting observation is that the robot team with speakers outperforms the other with respect to the mean performance (see Figure 8). In fact, cooperative behaviours occasionally take place in the robot team with speakers. Consequently, we conclude that the robots are able to find tasks after evolution, and the sound sensors and speakers work well. The reason cooperative behaviours do widely emerge is that the probability of EP1 and EP2 robots with high fitness partnering up is very low under “static-select”.

At the beginning of each epoch of each combined group from different subpopulations, the robots and tasks are placed in randomly generated positions. This explains why the best performance over 400 generations fluctuates significantly in Figure 7 and the mean performance is relatively steady in Figure 8.

![Figure 7. Evolution of the best performance with and without speaker.](image1)

![Figure 8. Evolution of the mean performance with and without speaker.](image2)
We carried out Exp 3 and Exp 4 in addition to Exp 2, and present the performance comparison in Figure 9. The results illustrate that the cooperative behaviours are successfully synthesized under “best-select” and “Pr-select”. Exp 3 and Exp 4 display very similar performance, which is significantly better than Exp 2. Moreover, Exp 3 outperforms Exp 4 in the early generations; however, the opposite is true in the later generations. This suggests that “best-select” is a good option for the early, unstable phase, and “best-select” is suitable for the later, stable phase.

Figure 9. Evolution of the best performance using different selection method.

Figure 10. Evolution of the best performance with different fitness function settings.

Figure 11. Evolution of the EPI’s best performance with and without gripper.
In Exp 3 and Exp 4, we observed the emergence of interesting evolution behaviours. Over the first few generations, robots do not even know how to find and execute tasks, let alone cooperate with another type of robot. Then, after robots learn how to accomplish tasks, some EP1 robots emit sounds, and other robots (including EP1 and EP2 robots) are attracted by the sound and help to carry out the tasks. EP1 robots find they do not benefit when they are attracted and engage in helping other robots; hence, they begin to neglect such sounds. In the end, EP1 robots concentrate on discovering task spots and attracting other robots, and EP2 robots are inclined to be attracted and help to execute tasks. In a word, every robot finally finds the appropriate niche in which it can make a useful contribution.

To test the hypothesis that each type of robot can synthesize its control strategy to accommodate its own capabilities even when all the robots share the same fitness function, we performed Exp 5 and compared performance to Exp 3 in Figure 10. The results illustrate that different options of fitness functions do not lead to significant changes in group fitness. It is important to note that $f_i$ in Exp 3 and $f_i^2$ in Exp 5 are compared, because $f_i^2$ has no meaning in comparison with $f_i$. We are always concerned with the total number of found and completed tasks.

The significance of the results in Figure 10 is the suggestion that using the same fitness function for different sub-groups is an alternative to using independent ones that are difficult to design. However, this conclusion requires further study before it can be widely applied.

As mentioned above, we introduce a simulated gripper to every robot in order to make it easier for robots to evolve the controllers. The expected result is obtained for EP1 robots as shown in Figure 11: EP1 robots are more likely to generate controllers to emit sounds to attract other robots to solve tasks with the help of gripper. In contrast, Figure 12 shows interesting and counterintuitive results with respect to the EP2’s fitness. The performances of EP2 with gripper and without gripper do not differ significantly. A possible reason could be that EP2 robots are able to synthesize effectively under both conditions. The results suggest that a good awareness of the position between tasks and robots may help robots to synthesize the desired controllers faster.

![Figure 12. Evolution of the EP2’s best performance with and without gripper.](image)

### 4.2 Verification of Controllers

Without loss of generality, the synthesized controllers from Exp 6 are used to demonstrate the controller’s stability. Although there are many definitions of stability, we are concerned with whether the outputs of the system remain bounded under different initial conditions. Three different cases are used in the demonstration, as shown in Table 3, where Case 1 is the original case used for evolving the controllers. Each case was run 10 times to determine the mean and standard deviation of the total number of completed tasks for achieving statistical significance. The start positions of robots and tasks can be identical or different in the 10 runs for each case. Note that the results can even vary for 10 runs with identical start positions, because each task will be placed again randomly to a new position after it is completed.

| Case | Number of EP1/EP2 robots | Number of tasks | Size of environment |
|------|--------------------------|----------------|---------------------|
| Case 1 | 5/5                      | 4              | 200-200             |
| Case 2 | 5/5                      | 4              | 400-400             |
| Case 3 | 10/10                    | 8              | 400-400             |

**Table 3.** List of cases to demonstrate the controller’s stability.

The results presented in Figure 13 suggest that the number of completed tasks is almost free from the impact of the settings of start positions of robots and tasks. It also
shows that the controllers are able to scale with the team size to ensure acceptable solution qualities. The standard deviation in each data set is very small. These data and the observation reveal that the controllers demonstrate relatively stable performance with different start position settings.

![Figure 13. Comparison of solutions between two different start position settings in three cases.](image)

In order to further evaluate the synthesized controllers from Exp 6, the aforementioned scenario has also been implemented and tested on a team of physical robots with different capabilities. The set of robots forming our application scenario is composed of two CCNT-A and two CCNT-B robots, as illustrated in Figure 14. Each robot is equipped with a Logitech QuickCam Pro 4000 camera, BSWA MPA 416 array microphones, infrared distance sensors, front and rear bumpers, and a beep speaker. For the sake of consistency, the capabilities of the CCNT-A robots are configured to be compatible with EP1, and the CCNT-B robots correspond to EP2. In the experiment, the robots are charged with three tasks in an outdoor environment. In contrast to the simulated environment, the tasks are removed from the environment manually and will not be inserted into the environment again when completed.

![Figure 14. Snapshot of a corner in the experimental environment with a CCNT-A robot (left), a CCNT-B robot (right), and a task (middle).](image)

As expected, all the robots spread out to find tasks at the beginning of the experiment. A CCNT-B robot found a task first, and went to solve the task by itself. The reason why the CCNT-B robot did not emit sounds to attract other robots to help it to solve the task is that it was capable of doing the work alone efficiently. Then, a CCNT-A robot found another task, and it preferred to emit sounds to attract others to help to execute the task. Two CCNT-B robots were both attracted by the sound and went to help the CCNT-A robot, while another CCNT-A robot was not attracted and continued looking for other tasks. Finally, the last task was found by the CCNT-A robot and completed with the help of CCNT-B robots.

The results were therefore consistent with those observed in the simulated environment. This verifies the feasibility of using evolution methods to synthesize controllers for physically heterogeneous MRS.

5. Conclusions

In this paper, we have considered synthesizing cooperative behaviours using artificial evolution for a team of physically heterogeneous robots. The method has been validated through a series of experiments with a team of e-puck robots. Satisfactory results have been obtained for the coordinated tasks. For future work, we will make a systematic comparison and evaluation of controller evolution under different conditions, and we also plan to reduce the computation cost by creating the necessary infrastructure to support distributed evolutionary computing.

6. Acknowledgment

This work was supported by the National Natural Science Foundation of China (61173177), the Defense Industrial Technology Development Program (B1420110149), the Zhejiang Provincial Natural Science Foundation of China (Y1090449), and the China Postdoctoral Science Foundation (2012M511369).
7. References

[1] S. Nolfi and D. Floreano (2000) Evolutionary robotics. Cambridge, Massachusetts: The MIT Press. 336 p.

[2] Y. C. Jin and Y. Meng (2010) Special Issue on Evolutionary and Developmental Robotics. IEEE Comput. Intell. Mag. 5(3): 9-9.

[3] J. M. McNamara, Z. Bart, L. Fromhage, et al. (2008) The coevolution of choosiness and cooperation. Nature. 451(7175): 189-192.

[4] T. Balch, F. Dellaert, A. Feldman, et al. (2006) How multirobot systems research will accelerate our understanding of social animal behavior. P. IEEE. 94(7): 1445-1463.

[5] J. Walker, S. Garrett, and M. Wilson (2003) Evolving controllers for real robots: A survey of the literature. Adapt. Behav. 11(3): 179-203.

[6] I. Harvey, E. Paolo, R. Wood, et al. (2005) Evolutionary robotics: A new scientific tool for studying cognition. Artif. Life. 11(1-2): 79-98.

[7] H. Lipson and J. B. Pollack (2000) Automatic design and manufacture of robotic lifeforms. Nature. 406(6799): 974-978.

[8] A. L. Nelson, G. J. Barlow, and L. Doitsidis (2009) Fitness functions in evolutionary robotics: A survey and analysis. Robot. Auton. Syst. 57(4): 345-370.

[9] G. Baldassarre, S. Nolfi, and D. Parisi (2003) Evolving mobile robots able to display collective behaviors. Artif. Life. 9(3): 255-267.

[10] M. Dorigo, V. Trianni, E. Sahin, et al. (2004) Evolving self-organizing behaviors for a swarm-bot. Auton. Robot. 17(2): 223-245.

[11] D. Floreano and L. Keller (2010) Evolution of Adaptive Behaviour in Robots by Means of Darwinian Selection. PLoS Biol. 8(1): 1-1000292.

[12] M. Potter, L. Meeden, and A. Schultz (2001) Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI). pp. 1337-1343.

[13] G. Capi (2007) Multiobjective Evolution of Neural Controllers and Task Complexity. IEEE T. Robot. 23(6): 1225-1234.

[14] S. Luke, C. Hohn, J. Farris, et al. (1997) Co-evolving soccer softbot team coordination with genetic programming. Proceedings of the First International Workshop on RoboCup, IJCAI-97, Nagoya, Japan, Springer-Verlag. pp. 398-411.

[15] A. Nelson, E. Grant, and T. Henderson (2004) Evolution of neural controllers for competitive game playing with teams of mobile robots. Robot. Auton. Syst. 46(3): 135-150.

[16] E. Uchibe and M. Asada (2006) Incremental coevolution with competitive and cooperative tasks in a multirobot environment. P. IEEE. 94(7): 1412-1424.

[17] P. Vadakkepat, X. Peng, B. Quek, et al. (2007) Evolution of fuzzy behaviors for multi-robotic system. Robot. Auton. Syst. 55(2): 146-161.

[18] H. Ishibuchi, R. Sakamoto, and T. Nakashima (2001) Evolution of unplanned coordination in a market selection game. IEEE T. Evolut. Comput. 5(5): 524-534.

[19] R. Watson, S. Ficici, and J. Pollack (2002) Embodied evolution: Distributing an evolutionary algorithm in a population of robots. Robot. Auton. Syst. 39(1): 1-18.

[20] M. Waibel, L. Keller, and D. Floreano (2009) Genetic team composition and level of selection in the evolution of cooperation. IEEE T. Evolut. Comput. 13(3): 648-660.

[21] J. Y. Weng, J. McClelland, A. Pentland, et al. (2001) Artificial intelligence - Autonomous mental development by robots and animals. Science. 291(5504): 599-600.

[22] Y. C. Jin and Y. Meng (2011) Morphogenetic Robotics: An Emerging New Field in Developmental Robotics. IEEE Trans. Syst. Man Cybern. Part C-Appl. Rev. 41(2): 145-160.

[23] M. Potter and K. De Jong (1994) A cooperative coevolutionary approach to function optimization. Proceedings of the International Conference on Evolutionary Computation and the Third Conference on Parallel Problem Solving from Nature, London, UK, Springer-Verlag. pp. 249-257.

[24] M. Potter and K. Jong (2000) Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evol. Comput. 8(1): 1-29.

[25] R. Wiegard, W. Liles, and K. De Jong (2001) An empirical analysis of collaboration methods in cooperative coevolutionary algorithms. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO). pp. 1235-1242.

[26] H. Blumenthal and G. Parker (2004) Co-evolving team capture strategies for dissimilar robots. Proceedings of AAAI 2004 Fall Symposium on Artificial Multiagent Learning, Washington.

[27] P. Duerr, C. Mattiussi, and D. Floreano (2010) Genetic Representation and Evolvability of Modular Neural Controllers. IEEE Comput. Intell. Mag. 5(3): 10-19.

[28] Teem: an open evolutionary framework. Available: http://lils2.epfl.ch/resources/teem/. Accessed 2012 JUL 11.