MONEE: Using Parental Investment to Combine Open-ended and Task-Driven Evolution

Nikita Noskov, Evert Haasdijk, Berend Weel, and A.E. Eiben

Vrije Universiteit Amsterdam, The Netherlands
n.k.noskov@gmail.com, e.haasdijk@vu.nl, b.weel@vu.nl, a.e.eiben@vu.nl

Abstract This paper is inspired by a vision of self-sufficient robot collectives that adapt autonomously to deal with their environment and to perform user-defined tasks at the same time. We introduce the MONEE algorithm as a method of combining open-ended (to deal with the environment) and task-driven (to satisfy user demands) adaptation of robot controllers through evolution. A number of experiments with simulated e-pucks serve as proof of concept and show that with MONEE, the robots adapt to cope with the environment and to perform multiple tasks. Our experiments indicate that MONEE distributes the tasks evenly over the robot collective without undue emphasis on easy tasks.

1 Introduction

The work presented in this paper is inspired by a vision of autonomous, self-sufficient robot collectives that can cope with situations unforeseen by their designers. An essential capability of such robots is the ability to adapt their controllers in the face of challenges they encounter in a hands-free manner, "the ability to learn control without human supervision," as [14] put it.

One approach to solve this issue uses evolution as a force for *adaptation*, rather than as "just" an algorithm for *optimisation*. This dichotomy has been noticed early in the history of evolutionary computing, [4]. Since then, these two attitudes have became dominant in different areas. Optimisation is the primary goal in Evolutionary Computing [6], while evolution as a driver of adaptation is typical in Artificial Life (ALife) [23]. In a common ALife setting, agents, possibly (simulated) robots, populate a world and the one that can cope with its environmental challenges will survive and reproduce. In systems like this, there need not be any objective function to be optimised, nor a centrally orchestrated evolutionary selection–reproduction loop. Instead, evolution is driven by a decentralised, asynchronous process of mate selection and reproduction and purely environmental selection that gives a reproductive advantage to well-adapted individuals. Such open-ended approaches are slowly finding their way into evolutionary robotics, e.g. the mEDEA algorithm[2] and Bianco and Nolfi's work [1].

Of course, an adapting robot collective must also serve the purpose of its designers: it must satisfy human preferences and tackle particular tasks. Evolutionary robotics has traditionally focussed exclusively on this latter aspect, employing evolution as a force for optimisation. Robots are set some specific task, their performance is measured through some objective function and a, typically centralised, evolutionary algorithm optimises robot behaviour accordingly.

In our vision, evolution serves two purposes: on the one hand to allow robots to adapt to the environment and to behave so that they can operate at all. On the other hand, evolution is a force to promote task-performance, where we interpret 'task' in a broad sense: it is any user-defined preference with a measurable level of compliance. It can be a direct task, like collecting rubbish (measured by the amount of rubbish cleared), but also more indirect, like energy efficiency (measured by battery lifetime). Combining these two (seemingly) contradictory roles of evolution is a generic, fundamental challenge that to our knowledge has to date not been tackled successfully.

As Jones and Mataríc note [7], collectively tackling tasks also entails a division of work: if, for instance, the swarm has two (sub-)tasks, it may be possible that all robots perform both tasks or that part of the swarm focusses on one task and the other robots tackle the second task. Therefore, an algorithm that enables our vision of an adaptive collective of robots should combine the adaptive and optimising facets of evolution as well as promote a good division of labour.

This paper introduces the MONEE (**M**ulti-**O**bjective a**N**d open-**E**nded **E**volution) algorithm that combines the open-ended and task-driven aspects of evolution. It is inspired by the open-ended algorithms described in [2] and [19].

MONEE allows the robot collective to optimise their behaviour to suit multiple tasks while distributing the tasks over that collective. It extends the open-ended approach as found in mEDEA [2] with a currency-based system where an individual can earn as well as spend credits. Please note that our idea of open-endedness in this case entails the lack of an explicit fitness function, whether our system adheres to the more general sense of open-endedness where evolution does not converge has not yet been tested. The main idea is that earnings are based on task-performance, while spendings are related to reproduction. Individuals accumulate credits through task performance - the better a robot performs a task, the more credits it earns for that task. When an individual puts its genome forward as a potential parent, it also passes information on its earnings for each defined task as a parental investment. Section 3 provides more detail on our implementation of the MONEE algorithm.

This paper provides a proof-of-concept for the MONEE algorithm. We simulate a collective of e-puck robots that are set multiple tasks to ascertain:

- if the robots adapt their behaviour to suit the tasks;
- if all tasks are performed equally well;
- how the system reacts to changing tasks.

2 Related Work

Open-ended and task-related evolutionary robotics Evolutionary Robotics has been widely studied since the early 1990s [15]. Initially, research focussed on individual robots, but since then substantial effort has been directed at evolution in larger numbers of interacting autonomous robots in swarms [20], research projects include for instance the Swarmanoid project [5]) or modular robots (e.g. M-tran[8]). In all these cases, evolution is used to achieve some fixed user-defined objective such as locomotion or explicit coordination.

Open-ended or objective-free evolution as well as self-replication have been studied in Artificial Life since Rasmussen's (1990) [16] and Ray's (1991) [17] work. Such research primarily investigates evolutionary dynamics in the absence of tasks, but as a result of implicit or environmental criteria that impact the ability to spread genomes through the population. This open-ended approach has gained interest from the evolutionary robotics community, for instance in Bianco and Nolfi's experiments with self-assembling organisms [1] and more recently in the mEDEA algorithm [2].

Open-ended approaches have been considered as a strategy to promote behavioural diversity in multi-objective settings by, for instance, Mouret and Doncieux [13]. Lehman and Stanley's novelty search [9] also embraces open-endedness to tackle elusive problems where a straightforward objective function leads to sub-optimal behaviour. These recent advances do define objective functions, though: the definition of novelty for Lehman and Stanley and the secondary objective for Mouret and Doncieux are ad-hoc, task-specific definitions of behavioural diversity that amount to tangential and creative redefinitions of the orginal objective function. Thus, such methods are not the completely objective-free approaches where survival and rate of procreation determine fitness rather than the other way around.

Parental Investment When animals reproduce they invariably invest time and energy in their offspring, for which Trivers coined the phrase parental investment [21]: "Parental investment covers any cost that a parent incurs in looking after an offspring, be it in gamete production, gestation or care after birth". Parental investment has been investigated in biology, and theories have been proposed on the evolutionary origins of the differentiation between sexes. These theories have also been verified in ALife settings, including experiments with robots [10,22,19]. In artificial life parental investment is often used to give the offspring a starting value of (virtual) energy [11,12,3,18] and a parent's energy level is often linked to task performance (e.g., agents tasked with eating grass to gather energy [3]). While these approaches benefit the offspring of good individuals, none of these approaches use parental investment as a method for parent selection. Distributed on-line evolutionary systems such as Watson et al's embodied evolution [24] sometimes employ (virtual) energy as a currency to determine parent selection [24,25].

3 MONEE: Multi-Objective And Open-Ended Evolution

At its core, MONEE is an adaptation of the mEDEA algorithm described by Bredèche et al. [2] and Schwarzer's artificial sexuality algorithm [19].

The robot lifecycle in MONEE consists of two phases: life and rebirth. The robots have a limited, fixed, lifetime during which they perform their actions; moving about, foraging, et cetera. When their lifetime ends, they enter the rebirth phase and become 'eggs': stationary receptacles for genomes that are transmitted by passing live robots. This rebirth phase also lasts a fixed amount of time, at the end of which the egg selects parents from the received genomes to create a new controller. The robot then reverts to the ‘life’ role with this new controller. Thus, robots (or rather, their controllers) can procreate by transmitting their genome to eggs, and the more eggs a robot inseminates, the more chances it has for procreation. Because the transmission of genomes is continuous and at close range (e.g. through infrared), the more a robot moves about the arena, the better its chances of producing offspring. This aspect of MONEE is clearly open-ended: there is no calculated performance measure that defines the chances of being selected as parent, there is no task. Only the environment dictates what robots may or may not become parents.

To add task-driven parent selection to this basic evolutionary process, we use parental investments. During their lifetime, robots amass credits by performing tasks. For instance, a robot could get one credit for every piece of ore it collects, one for successfully solving some puzzle, and so on. If multiple tasks are defined, the robots maintain separate counts for the credits awarded for each task, for instance one counter for the pieces of ore collected and another one for the number of puzzles solved. When a robot inseminates an egg, it passes these numbers along with the genome and the egg uses that information to select parents when it revives.

When a robot’s egg phase finishes, it compares the parental investment for each genome it has received. To enable this comparison across task, the egg calculates an exchange rate between tasks. This ensures that genomes that invest in tasks for which few credits are found overall (presumably hard tasks) are not eclipsed by genomes that favour easier tasks. The pseudo-code in algorithm 1 details this auction scheme.[1]

The parental investments relate task performance to reproductive success: besides the open-ended goal of ‘merely’ transmitting genomes to eggs, robots must also become proficient at the defined tasks for these genomes to be selected. The more proficient a robot is at a task, the higher its chances of procreating. The comparison of investments across multiple tasks introduces an exchange rate between the earnings per task: the more common credits are for a particular task, the less their worth and vice versa. Thus, parent selection becomes a marketplace for skills and features that the user requires. Users can influence this economy to prioritise tasks, for instance by setting a premium on investments related to a particular task that the user deems more urgent than others. This system naturally caters for multi-objective approaches and allows the user to prioritise tasks in a straightforward manner.

4 Experiments

To investigate the MONEE algorithm, we implement it in a scenario with simulated e-pucks in a simple 2D simulator.[2] This scenario places 100 robots in an arena roughly 330 robots wide, with a number of obstacles (depicted in the lower

[1] Note, that our implementation uses roulette wheel selection and only mutation only, so a single parent is selected. These are incidental design choices: MONEE does not preclude the use of other selection schemes and/or recombination operators.

[2] RoboRobo, `https://code.google.com/p/roborobo/`

Algorithm 1: Selecting a parent based on parental investments

```
for every defined task do                                              // total credits
    for every received genome do
        $credits_{task} \leftarrow credits_{task} + genome.credits_{task}$
    end for
    $credits_{overall} \leftarrow credits_{overall} + credits_{task}$
end for

for every defined task do                                              // exchange rate per task
    $rate_{task} \leftarrow \frac{credits_{overall} + num_{tasks}}{credits_{task} + 1}$
end for

for every received genome do                                           // parental investment per genome
    for every defined task do
        $genome.rating \leftarrow genome.rating + (genome.credits_{task} \cdot rate_{task})$
    end for
end for

$parent \leftarrow roulettewheel\_selection(received\ genomes)$        // select and mutate
$child \leftarrow mutate(parent)$
$reactivate(child)$                                                    // revive
```

right of Fig. 1) and defines seven concurrent foraging tasks. Concurrent foraging is a variation of regular foraging where the arena is populated by multiple types of objects to be collected [7], rather than just a single resource. In our case, just as in [7], these objects are pucks of different colours, and the collection of each different colour is a different task.

We use seven differently coloured pucks, defining seven similar but different tasks. The colours are *SteelBlue*, *OrangeRed*, *LimeGreen*, *Indigo*, *SeaGreen*, *SandyBrown* and *Siena*. The pucks are spread in the environment according to a number of gaussian distributions as indicated in Fig. 1. As can be seen from the distributions, some colours (like *Indigo* and *SandyBrown*) are placed very compactly at the corners of the arena, which makes gathering those a more specialist proposal than for instance *SteelBlue* pucks, which can be found scattered across the entire arena. The number of pucks per colour varies between 25 and 150, also indicated in Fig. 1. When placing pucks we make sure that they do not overlap with existing pucks, robots or other obstacles.

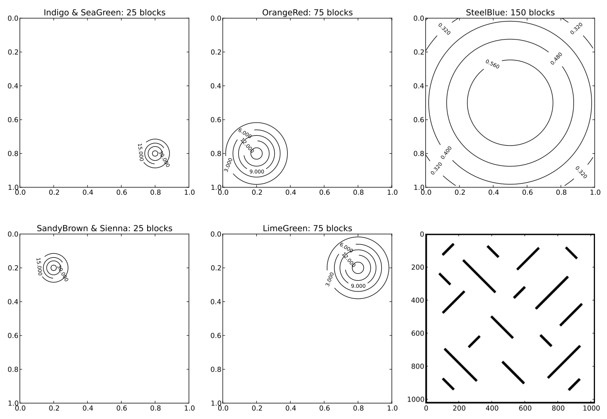


Figure 1. Distribution of pucks of various colors and obstacles present in the arena.

Robots gather these pucks simply by driving over them, and as soon as a robot has gathered a puck it is immediately removed: the robots do not have to transport the puck to a particular region. A replacement puck of the same colour is then randomly placed in the arena, taking the appropriate distribution into account, to allow the experiment and evolution adequate time.

Each robot is controlled by a single-layer feed forward neural network which controls its left and right wheels. The inputs for the neural network are the robot's sensors: a robot has 8 sensors for each type of puck, as well as 8 sensors to detect environmental obstacles and other robots. The layout of these sensors is that of a standard e-puck's infrared sensors: four face forward, two to the sides and two face backwards. Because the robots have separate sensors for each type of puck and the network only has direct connections from input to output, the task of collecting each type of puck –although very similar– needs to be learned completely separately.

The robot's genome directly encodes the neural network's weights (8 types of sensor × 8 sensors × 2 outputs plus 2 bias connections plus 4 feedback (current speed and current rotation to either output) = 134 weights) as an array of reals.

As specified by the MONEE algorithm, the robots alternate between periods of explorative block gathering and motionless genome reception. To prevent synchronised cycles among the robots, we add a small random number to each robot's fixed lifetime. This forces desynchronised switching between life and rebirth even though our runs start with all robots perfectly in sync at the first time-step of their lifetime.

At the end of the egg phase offspring was created by selecting a parent from the received genomes according to the parental investment and mutating it using gaussian perturbation with a single, fixed mutation step size $\sigma = 1$.

To investigate the response of this system to dynamically changing tasks, we radically change the distribution of pucks during the runs: halfway through the run, at 1 million iterations, all distributions generate pucks of a single colour only: only *SteelBlue* in half the runs, only *Indigo* in the others (i.e., either a common or a rare task remains).

Due to time constraints, we were only able to conduct a limited number of runs. Therefore, it is not feasible to provide meaningful statistics on the exact level of performance. The data does suffice, however, to indicate the potential of the MONEE approach and to analyse some of the dynamics of a population of robots that use MONEE to adapt their behaviour.

5 Results & Analysis

Irrespective of the foraging tasks, the robots must cope with their environment to be able to procreate: they must at the very least develop controllers that drive around the arena to inseminate eggs. Thus, the environment implies that the robots should move around. The robots should also avoid obstacles, even though this is not specified as an explicit task. If they do not, the time they spend trying to drive through an obstacle cannot be spent spreading their genome, limiting their chances of creating offspring. Therefore, we measure the number of collisions between robots and obstacles to gauge the level of adaptation to the environment. Figure 2 shows the total number of collisions for the whole collective over time. The number of collisions is aggregated over 1000 time steps. The number of collisions decreases with time. Even though the decrease is not spectacularly steep, it is a clear indication that controllers do adapt to the environment without any specifically set goal.

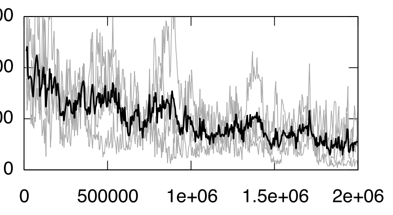


Figure 2. Number of collisions over time. The grey lines denote individual runs, the black line shows the average over the four runs.

Of course, we did specify the foraging tasks for the robots. Figure 3 shows the number of pucks harvested against time, here too the number of pucks gathered is aggregated over 1000 time steps. Clearly, the robots do evolve effective foraging behaviour: the number of collected pucks per time unit steadily increases throughout the runs. Changing the environment so that only pucks of a single colour are generated in most cases actually leads to a slight increase in the total number of pucks gathered. This seems to indicate that the robots in those runs do not specialise in a particular task. Possibly, when only a single colour remains, they are not distracted by pucks of a different colour and they therefore forage more effectively.

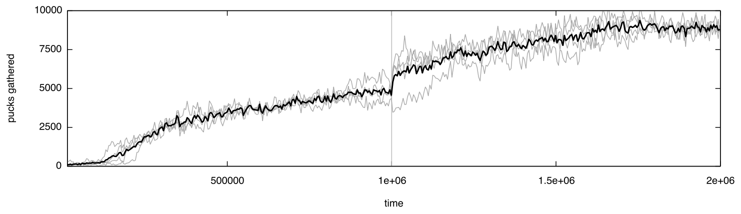
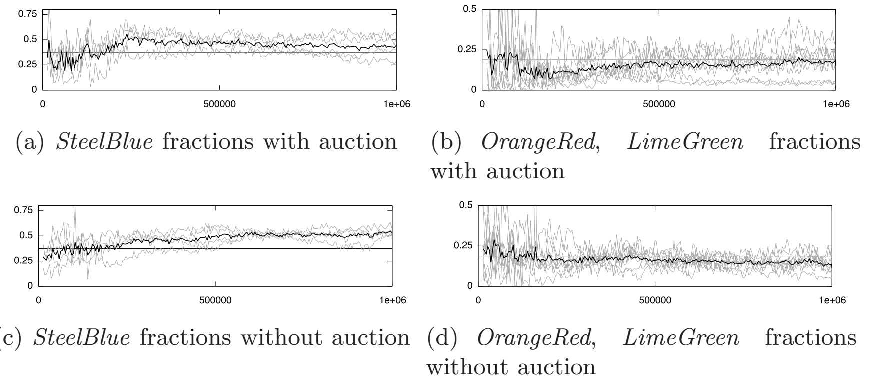


Figure 3. Number of pucks gathered over time. The grey lines denote individual runs, the black line shows the average over the four runs. The vertical line indicates the moment where the task changes and all pucks become a single colour.

To asses the efficacy of MONEE's currency scheme to distribute tasks, we also ran our experiments with the exchange rate mechanism turned off. In this case, a genome's chance of selection is related purely to the number of pucks that it collected without any consideration for their colour. Therefore, genomes that encode harvesting behaviour for rare colours are at a disadvantage. Figure 4 compares the fraction of *SteelBlue*, *OrangeRed* and *LimeGreen* pucks out of all gathered pucks with and without the exchange rate mechanism. The plots only show the first million time-steps because after that only a single colour remains as described above. With the exchange rate mechanism turned on (Figs. 4(a) and 4(b)). In both cases, the fraction of pucks gathered tends towards the actual fraction of pucks available: the trend in Fig. 4(a) decreases to the natural ratio of 0.375, indicating that the easiest task of collecting ubiquitous pucks is balanced with the harder task of collecting rarer pucks. Similarly, the fractions of *OrangeRed* and *LimeGreen* pucks in Fig. 4(b) slowly seem to increase to level off at the natural ratio of 0.1875. Figures 4(c) and 4(d) show a different picture. Without the exchange rate mechanism, the simple task is increasingly favoured as shown by the continuing rise of the *SteelBlue* fraction. This is at the expense of collecting rare colours, as indicated by the decreasing trend in Fig. 4(d).



(a) *SteelBlue* fractions with auction

(b) *OrangeRed*, *LimeGreen* fractions with auction

(c) *SteelBlue* fractions without auction

(d) *OrangeRed*, *LimeGreen* fractions without auction

Figure 4. Fraction of gathered pucks that are *SteelBlue* and *LimeGreen* for runs with and without the exchange rate auction mechanism turned on. Light grey plots for individual runs, black lines show the average over the runs. Horizontal black lines indicate the fraction of all pucks for the respective colours (0.375 and 0.1875).

To verify the decrease in collecting rare pucks, we ran a more extensive experiment. To isolate this claim more purely we used only 2 colours, red and blue, in a 1:3 ratio, i.e. 50 blue pucks and 150 red pucks. This amounts to a natural collection ratio of 0.25 for blue pucks. All other experimental parameters were kept the same as the previous experiments, except that we ran 64 repeats.

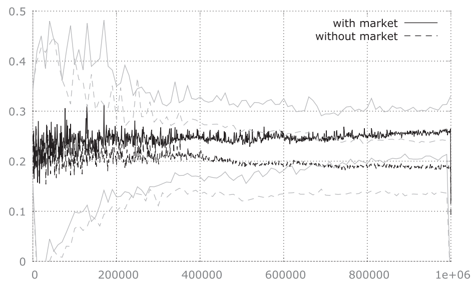


Figure 5. Market fraction for verification runs. Grey lines indicate standard deviation

Figure 5 shows the market fraction of blue pucks gathered, with and without market. As you can see the ratio for blue pucks gathered *with* market trends towards the natural ratio of 0.25, while the ratio for blue pucks gathered *without* market steadily drops over the course of a run. Although the difference in ratio without market is not significant at time step $1e^6$ (at least, not for 64 repetitions), the downward trend would seem to continue if the experiment was extended.

6 Conclusions & Further Research

We have introduced the MONEE algorithm as a tool to combine the open-ended and task-driven facets of evolutionary robotics. As a proof of concept, we ran experiments where robots have to move about an obstacle-strewn arena while concurrently foraging seven types of puck. The robot controllers were laid out so that this amounts to having to learn seven distinct tasks.

MONEE allows the robots to learn to cope with their environment as shown by the decreasing frequency of collisions. It also drives task-driven adaptation: the robots become increasingly proficient at the gathering tasks and a momentous change where six of the tasks disappear has no ill effect on the collective task performance. The exchange rate mechanism allows effective division of the tasks over the collective without favouring easier tasks at the cost of harder ones.

We emphasise once again that these results are based on a very limited number of runs. Nonetheless, they provide a good indication of algorithm behaviour, enough at least to show that the MONEE algorithm opens a promising avenue of further research. We are of course planning to conduct further experiments to provide solid statistical foundations for the indications we show in this paper. Moreover, we are keenly interested in researching the intricacies of the economy that results from the exchange rate mechanism in the face of more dynamic changes in task composition as well as the results of enforcing some level of task specialisation.

References

1. R. Bianco and S. Nolfi. Toward open-ended evolutionary robotics: evolving elementary robotic units able to self-assemble and self-reproduce. *Connection Science*, 4:227–248, 2004.
2. N. Bredeche, J.-M. Montanier, W. Liu, and A. F. Winfield. Environment-driven distributed evolutionary adaptation in a population of autonomous robotic agents. *Mathematical and Computer Modelling of Dynamical Systems*, 18(1):101–129, 2012.
3. M. Burtsev, V. Red'ko, and R. Gusarev. Model of evolutionary emergence of purposeful adaptive behavior. the role of motivation. In J. Kelemen and P. Sosík, editors, *ECAL*, LNCS 2159, pages 413–416. Springer, 2001.
4. K. DeJong. Are genetic algorithms function optimizers? In R. Männer and B. Manderick, editors, *Proceedings of the 2nd Conference on Parallel Problem Solving from Nature*, pages 3–13. North-Holland, Amsterdam, 1992.
5. M. Dorigo, D. Floreano, L. M. Gambardella, F. Mondada, S. Nolfi, T. Baaboura, M. Birattari, M. Bonani, M. Brambilla, A. Brutschy, D. Burnier, A. Campo, A. Christensen, A. Decugnière, G. Di Caro, F. Ducatelle, E. Ferrante, A. Förster, J. Guzzi, V. Longchamp, S. Magnenat, J. Martinez Gonzales, N. Mathews, M. Montes de Oca, R. O'Grady, C. Pinciroli, G. Pini, P. Rétornaz, J. Roberts, V. Sperati, T. Stirling, A. Stranieri, T. Stützle, V. Trianni, E. Tuci, A. E. Turgut, and F. Vaussard. Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics & Automation Magazine*, page in press, 2012.
6. A. E. Eiben and J. Smith. *Introduction to Evolutionary Computing*. Springer, Berlin Heidelberg, 2003.
7. C. Jones and M. Mataric. Adaptive division of labor in large-scale minimalist multi-robot systems. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1969 – 1974, oct. 2003.
8. H. Kurokawa, E. Yoshida, K. Tomita, A. Kamimura, S. Murata, and S. Kokaji. Self-reconfigurable m-tran structures and walker generation. *Robotics and Autonomous Systems*, 54(2):142–149, 2006.

9. J. Lehman and K. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011.
10. S. Mascaro, K. Korb, and A. Nicholson. An alife investigation on the origins of dimorphic parental investments. In H. A. Abbass, T. Bossomaier, and J. Wiles, editors, *Advances in Natural Computation*, volume 3, pages 171–185, 2005.
11. F. Menczer and R. Belew. Latent energy environments. In *Santa Fe Institute Studies In The Sciences Of Complexity-Proceedings Volume-*, volume 26, pages 191–210, 1996.
12. F. Menczer, W. Willuhn, and R. Belew. An endogenous fitness paradigm for adaptive information agents. In *CIKM Workshop on Intelligent Information Agents*. Citeseer, 1994.
13. J.-B. Mouret and S. Doncieux. Encouraging behavioral diversity in evolutionary robotics: an empirical study. *Evolutionary Computation*, 20(1):91–133, 2012.
14. A. L. Nelson, G. J. Barlow, and L. Doitsidis. Fitness functions in evolutionary robotics: A survey and analysis. *Robotics and Autonomous Systems*, 57(4):345 – 370, 2009.
15. S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, Cambridge, MA, 2000.
16. S. Rasmussen, C. Knudsen, R. Feldberg, and M. Hindsholm. The coreworld: Emergence and evolution of cooperative structures in a computational chemistry. *Physica D: Nonlinear Phenomena*, 42(1):111–134, 1990.
17. T. S. Ray. Is it alive or is it GA? In R. Belew and L. Booker, editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 527–534. Morgan Kaufmann, San Francisco, 1991.
18. M. Scheutz and P. Schermerhorn. Predicting population dynamics and evolutionary trajectories based on performance evaluations in alife simulations. In H.-G. Beyer and U.-M. O'Reilly, editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2005)*, pages 35–42. ACM, ACM, 2005.
19. C. Schwarzer, C. Hösler, and N. Michiels. Artificial sexuality and reproduction of robot organisms. In P. Levi and S. Kernbach, editors, *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*, pages 384–403. Springer-Verlag, May 2010.
20. V. Trianni. *Evolutionary swarm robotics: evolving self-organising behaviours in groups of autonomous robots*, volume 108. Springer, 2008.
21. R. Trivers. Parental investment and sexual selection. In B. G. Campbell, editor, *Sexual Selection and the Descent of Man*, chapter 7, pages 136–179. Biological Laboratories, Harvard University, 1972.
22. J. Ventrella. Genepool: Exploring the interaction between natural selection and sexual selection. *Artificial Life Models in Software*, pages 81–96, 2005.
23. M. Ward. *Virtual Organisms: The Startling World of Artificial Intelligence*. Pan Books, 2010.
24. R. A. Watson, S. G. Ficici, and J. B. Pollack. Embodied evolution: Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39(1):1–18, April 2002.
25. S. Wischmann, K. Stamm, and F. Wörgötter. Embodied evolution and learning: The neglected timing of maturation. In F. Almeida e Costa, editor, *Advances in Artificial Life: 9th ECAL*, LNAI 4648 , pages 284–293. Springer-Verlag, Lisbon, Portugal, September 10–14 2007.