HAL
open science

Environment-driven Open-ended Evolution with a Population of Autonomous Robots

Nicolas Bredeche, Jean-Marc Montanier

To cite this version:

Nicolas Bredeche, Jean-Marc Montanier. Environment-driven Open-ended Evolution with a Population of Autonomous Robots. Evolving Physical Systems Workshop, 2012, East Lansing, United States. hal-00731422

HAL Id: hal-00731422

https://inria.hal.science/hal-00731422v1

Submitted on 12 Sep 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Environment-driven Open-ended Evolution with a Population of Autonomous Robots

Nicolas Bredeche[1], Jean-Marc Montanier[1]

[1] TAO, LRI ; Univ. Paris-Sud, CNRS, INRIA ; F-91405 Orsay, France
firstname.name@lri.fr

Abstract

This paper summarizes work done since 2009 on running swarm of autonomous robots with Environment-driven Distributed Evolutionary Adaptation algorithms (EDEA). The motivation behind this work is to design algorithms that can cope with unknown environmental pressure, that is to learn (through evolution) efficient survival strategies in order to address a priori unknown constraints existing in the environment. We are concerned with a fixed-size population of autonomous robotic agents facing unknown, possibly changing, environments, which may require the robots to evolve particular behaviors such as cooperative behaviors, specialized behaviors, etc. We describe a particular flavor of EDEA, termed minimal EDEA (mEDEA), and summarize results obtained so far, both with real robots and in simulation. This algorithm is shown to be both efficient in unknown environment and robust with respect to abrupt, unpredicted, and possibly lethal changes in the environment. Moreover, the ability of mEDEA to evolve various types of behavior, including cooperative and specialized behaviors, is summarized.

Introduction

Achieving Self-sustainable Autonomous Robotics is one challenging task that is yet to be fully addressed. Designing such systems is motivated by applications where humans are no longer able to craft the system to the environment, either due to environmental complexity or lack of prior knowledge. Examples of applications can be found whenever unpredictable situations arise during the actual use of a robotic systems: deployment of mobile sensor network devices with limited communication abilities (Hauert et al., 2009), self-reconfigurable robots for highly constrained environments (Yim et al., 2007; Shen et al., 2006), radiation shielding and site clearing (Engwirda, 2006) or even space exploration, as illustrated by NASA's interest in self-sustainable robotic systems (Colombano and Shen, 2006).

In a different area, Open-ended Evolution has drawn quite a lot of attention in the field of Artificial Life, with Tierra (Ray, 1991) and its followers. While the scientific objectives are related to the simulation of open-ended evolutionary processes, these approaches may also be studied from the perspective of algorithmic design towards a practical problem: how to achieve open-ended evolution in the real world using artificial systems such as robots.

In this paper, we consider a population of autonomous robots (i.e. fixed population size) with limited local communication capability that should be able to learn how to achieve self-sustainability (i.e. survive) in an unknown, possibly dynamic, environment. The work described hereafter is concerned with the design of distributed algorithms to endow a swarm of autonomous robots with the capability to change its behaviour on-the-fly depending on the environmental conditions at hand. We refer to this particular class of algorithms as *Environment-driven Distributed Evolutionary Adaptation*, or **EDEA** for short.

In EDEA, evolution occurs among a population of (evolving) behavioural strategies (encoded within genomes), considering the population of robots as the actual resource to share (cf. section 2 for details) - as with Dawkins selfish gene metaphor, a survival strategy that succeeds in generating many offsprings (i.e. spreading in the population) is considered a fitted strategy. As the population is limited in size, strategies compete with one another to share such resource. In previous works, we addressed open-ended evolution in a population of robotic agents under various environmental pressures, such as changing environments (Bredeche and Montanier, 2010), environments requiring the emergence of altruistic behaviors (Montanier and Bredeche, 2011), with both simulated and real robots (Bredeche et al., 2012) (cf. figure 1 for an illustration).

This paper is organized as follows: the mEDEA algorithm is described and discussed in Section 2. In section 3, results from experiments using 20 e-puck robots are summarized, including some considerations on the robustness of the algorithm in the real world. Then, Section 4 gives a summary of results and lessons learned so far. Section 5 presents a discussion on this class of algorithms, and highlight similarities and differences with closely related fields, with a particular emphasis on Embodied Evolutionary Robotics. Finally, the last Section sketches a tentative roadmap for future investigations and general considerations.

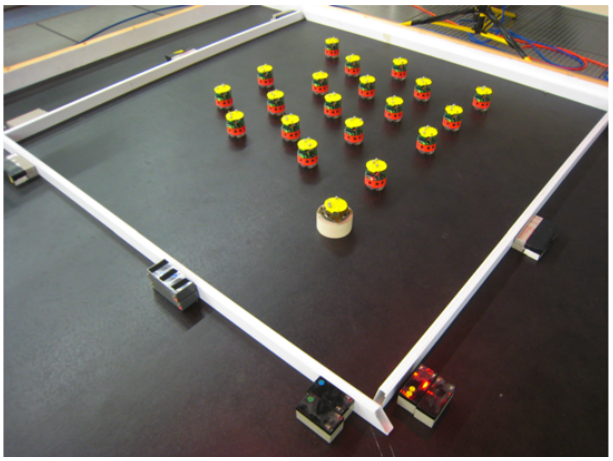

Figure 1: Environment-driven self-adaptive swarm of robots - experiment with 20 e-puck mobile robots (details in (Bredeche et al., 2012))

mEDEA: A Minimal EDEA

The **mEDEA** algorithm, as in *minimal* EDEA, was initially introduced in (Bredeche and Montanier, 2010). It performs as an evolutionary adaptation algorithm that can be distributed over a population of robotic agents (i.e. each agent in the population runs the same algorithm, but carries different genomes). mEDEA has been designed from the start with on-board robotic implementation as a primary motivation, and has been tested in various setups and environments so far.

Figure 2 provides an illustrative example of how mEDEA works (see Annex for a description of the algorithm). Each agent/robot contains an *active* genome, which (indirectly) controls the agent's behaviour, and a *reservoir of stored genomes*, which is empty at first. At each time step, each agent *broadcasts* in a limited range a *slightly* mutated copy of its active genome (gaussian mutation) and stores genomes received from neighbours, if close enough (and if not already stored). At the end of a generation (i.e. a pre-defined number of time steps), each agent "forgets" its active genome and *randomly* picks one genome from its reservoir of stored genomes (if not empty). Then the reservoir is emptied, and a new generation starts. This algorithm is duplicated within each agent in the population, even though agents' behaviours differ depending on each agent's current active genome.

As in other algorithm for open-ended evolution, selection pressure occurs at the population level (the more a genome spreads itself, the higher the probability it will generate offsprings) rather than at the individual level (random sampling). Then, genomes survive only through spreading (as an active genome is automatically deleted locally at the end of a generation) and individual may get better over time as conservative variations generate new candidates that explore

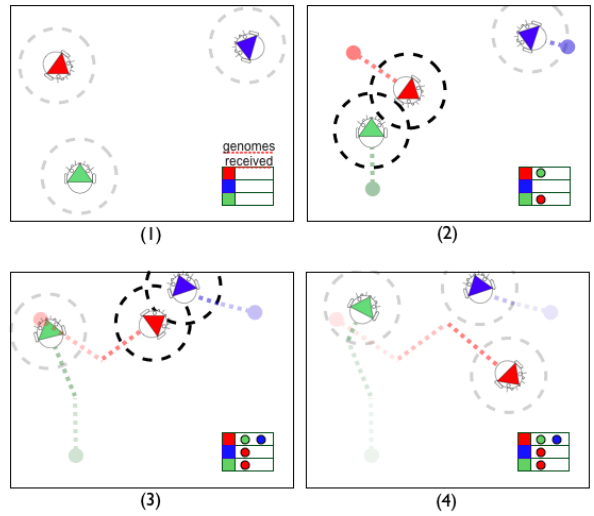

Figure 2: The mEDEA algorithm: a simplified illustration. (1): generation starts, genome lists are empty; (2) and (3): robots move around (each robot is controlled by its own active genome) and exchange genomes when close enough ; (4): generation ends - the red genome spread more and thus have higher probability of being selected (in this case, probability is indeed $p = 1$ in two robots while the two other genomes only get $p = 0.5$ in one single robot. Note that the next generation will contain slightly mutated copies of the original genomes

alternative (but closely related) behavioural strategies. As with mEDEA, the most important characteristics for algorithms that belong to the EDEA class is that **no** fitness function is ever formulated, making it possible to address truly open-ended environments.

In addition to this description, several additional mechanisms are implemented that deal with specific issues:

- While only a mutation operator is used in mEDEA, other variation operators may be used as well. As with other Evolutionary Algorithm, conservative variations (i.e. generating offsprings in the *vicinity* of their parents, in term of fitness landscape) are preferable. For example, we have also experimented with various cross-over operators, including horizontal gene transfer operators.

- A minimum of 3 agents is required in order to ensure that selection pressure is possible (but not guaranteed). Indeed, two agents would only exchange genomes, implying deterministic rather than random selection. In practical, larger populations are preferred (cf. next sections).

- There is no guarantee whatsoever that a given robotic agent will be able to meet with at least one single other agent during its lifetime. This would result in the "death" of this robot, as no new genome is available to be picked. Several workarounds may be applied, however

it is mandatory to avoid introducing a pressure towards unwanted selection pressure (e.g. reusing the previous genome should be avoided as it would result in providing a selective advantage towards "getting lost" behaviours). Of course, global extinction is always possible, and depends on various parameters, including the interaction between the environment and the population (cf. next sections for practical solutions).

The mEDEA algorithm provides an evolutionary adaptation mechanism in open-ended environment, but does not provide a control function. The actual control of the agent behavior shall be performed by a dedicated controller whose parameters are determined from the genome. In other words, the mEDEA algorithm provides evolutionary adaptation by tuning the control architecture, which can be (and is, in some of our experiments) a Multilayer Perceptron that maps the sensory inputs to motor outputs (e.g. 8 infrared sensors and left/right motors of a robot resembling the e-puck mobile robot) and whose weights are encoded within the genome.

A population of e-pucks running mEDEA

This section summarizes several experiments using several e-puck robots that have already been described in (Bredeche et al., 2012), and provides new details regarding behavioural strategies as well as considerations with respect to real world implementation.

We implemented the mEDEA algorithm within a population of e-puck robots, provided by the Bristol Robotics Lab. Each e-puck is equipped with 8 Infra-Red (IR) proximity sensors, 3 microphones, 1 loudspeaker, 1 IR remote control receiver, a ring of 9 red LEDs + 2 green body LEDs, 1 3D accelerometer and 1 CMOS camera. Moreover, each e-puck is extended with a linux board (Liu and Winfield, 2011), which makes it possible to implement the algorithm in $C/C++$.

A population of up to 20 e-pucks robots is placed in a $280\ cm * 230\ cm$ empty arena. Limited-range communication (approx. 20cm) is emulated by using a WiFi network combined with a Vicon™tracking system[1]. All experiments described in this section lasted from 30 min to 45 min, and at least 25 generations (each generation lasts approx. 1 min and 10 sec, corresponding to 400 controller update steps – this time is sufficient for a controller to cross the whole area, and therefore meet most of the robots). The refresh rate for a robot controller is limited to 5 updates per second due to various technical limitations of the setup. Robots are not globally synchronized, implying the generation number may vary from one robot to another (asynchroneous evolution).

An additional object is introduced into the arena and is referred to as the **sun**: each robot knows the sun's relative orientation and distance thanks to the Vicon™system - being close to, or away from the sun provides *a priori* **no** selective advantage. The experimenter may arbitrarily change the sun's location from time to time during the course of the experiment, switching the sun location from one end of the arena to the other. The goal in this experimental setup is to study how a population of robots may evolve towards a behavioural strategy for survival within an environment with one particular, *a priori* useless, singularity.

Figure 3 illustrates one typical run (out of 8 independent runs with 20 robots). Figure 4 details this run and shows the corresponding trajectories. Each image represents a 4 minutes time span (approx. 3 generations), during which each robot location is recorded every 1 sec (the sun location is illustrated as a yellow circle). The experiments is started with a sun located in the south-east quadrant of the environment. While robots are initially displaying various simple behaviours (turning around, heading forward, standing still, etc.), one strategy quickly evolves as a large part of the population drive to and remain at the singularity point (i.e. the sun). Though this strategy is not the only one to be observed, it is displayed by more than half the population. After some time (approx. 25 min), the sun is moved to some other location by the experimenter (i.e. the two-legged primate in figure 3, image no.3). This makes it possible to witness a sun-following behavior that was evolved earlier. Note that as a large part of the population travel towards the new sun location, these robots may also encounter other robots – possibly wiping out less represented alternative strategies, and ending with even more robots displaying the same behaviour (cf. full details can be found in (Bredeche et al., 2012)).

These experiments revealed a number of technical issues unfamiliar to experiments in simulation, as can be expected when trying to cross the so-called reality gap (Jakobi et al., 1995). In the present setup, the following issues where experienced:

- Proximity sensors are unreliable: the low quality of the infra-red sensors makes it very difficult to detect obstacles and/or other robots (with a binary positive response only for distances under 1cm). Also the e-puck body is more or less transparent, making it almost invisible in some cases. Adding a coloured plastic skirt to the robots only partly solves the problem as the proximity sensors are occasionally blinded by the skirt. As a consequence, proximity sensor values must often be disregarded because of their unreliability.

- Colliding robots are regularly unable to send/receive genomes to their neighbours. This is due to our particular setup where local communication is emulated using the Vicon™system for computing the local communication network based on distance between robots (the Vicon™fails to identify colliding robots). While this approach was originally motivated by the absence of local robot-robot communication (and lack of time[2] to develop

[1] http://www.vicon.com

[2] Implementing and running the experiments was completely done during a one-month stay at BRL.

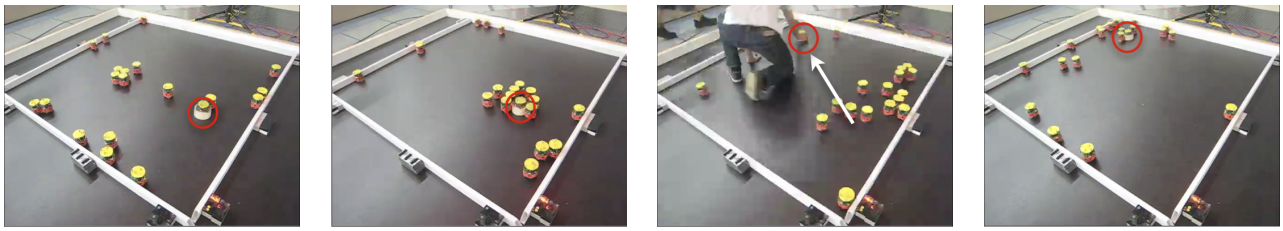

Figure 3: A typical experiment with 20 e-pucks running mEDEA. These four images are extracted from a sequence available at this address: `http://www.youtube.com/watch?v=_ilRGcJN2nA`

IR-based communication libraries), it ended up as quite a problem as robots could not participate in the evolutionary adaptation process during collisions. In practice robots were occasionally lost from the emulated communication network but always recovered when moving away from one another.

- On-board processing is slow: the combination of on-board computation with limited hardware and the particular setup for emulating local communication has a negative impact on speed of execution. In fact 5 updates/sec was observed, with often asynchronous updates of sun distance and orientation from the tracking system (i.e. out-of-date information, or missing opportunities).

While simulation and real world experiments do differ as expected, the MEDEA algorithm proved to be remarkably robust to the reality gap, as it manages to deal with all of the issues outlined above and still demonstrates interesting behaviours that manage to survive in the environment. As a matter of fact, the most critical issue (in simulation and in the real world) is the size of population: larger populations display more stable behaviours, while very small population (e.g. < 20 robots in this context) are prone to extinction. This was confirmed by results from some previous experiments using less than 10 robots: results were difficult to analyze (few runs converged towards behavioural consensus, and some did not even converge at all (i.e. extinctions)) - this was also experimentally confirmed using extensive simulation (cf. (Bredeche et al., 2012)).

Lessons learned so far

The mEDEA algorithm has been studied in various contexts so far: given an *a priori* unknown environment with specific properties, the goal is (a) to study the evolutionary dynamics of mEDEA (or variations) to achieve survivability, (b) to identify the particular strategies that have been evolved, (c) optionally, to modify the algorithm so as to increase the ability for the whole population to survive in this particular setup. In the following, questions addressed so far, as well as lessons learned, are briefly summarized:

- Evolving homogeneous behavior (Bredeche et al., 2012): extending what was presented in the previous section, we studied how consensus towards a specific behaviour could be evolved and spread over the population. In these experiments, the importance of population size and mating opportunities (which depends on population size, radius of communication and environment size) was highlighted. It was also shown that as both population size and mating opportunities increase, a single behavioural strategy would dominate within the entire population.

- Robustness to environmental changes (Bredeche and Montanier, 2010) : In this work, the ability of mEDEA was experimentally tested with respect to changing environmental conditions. It was shown that a population of (virtual) robots running mEDEA was able to recover from a sudden change in the environment towards much more challenging conditions. While survival rate was largely affected by such a change, mEDEA quickly recovered by evolving behaviours fitted to the new environment at hand, without having to be modified or notified by the user.

- Learning altruistic behaviour (Montanier and Bredeche, 2011): mEDEA was tested within a particular environment inspired from the Tragedy of the Commons (Hardin, 1998). In such environments, altruistic behaviours (i.e. choosing to share part of what one could keep for itself) is required in order for the whole population to survive. In this setup, it was shown that mEDEA naturally evolves altruistic behaviours over the population if environmental conditions become too challenging. Also, it was shown that the algorithm globally converges towards a trade-off between altruism and selfishness to ensure survival of the majority of the population (i.e. sharing as much as possible as long as it does not impact one's own survival chances).

- Specialization through speciation (on-going work): mEDEA is currently studied with respect to its ability to converge towards specialized sub-populations whenever it is required (e.g. foraging two energy resources, each available in limited quantity and each requiring specific skills). In this scope, we consider behaviour specialization through speciation (i.e. generalists cannot be evolved). Results show that while protecting species can

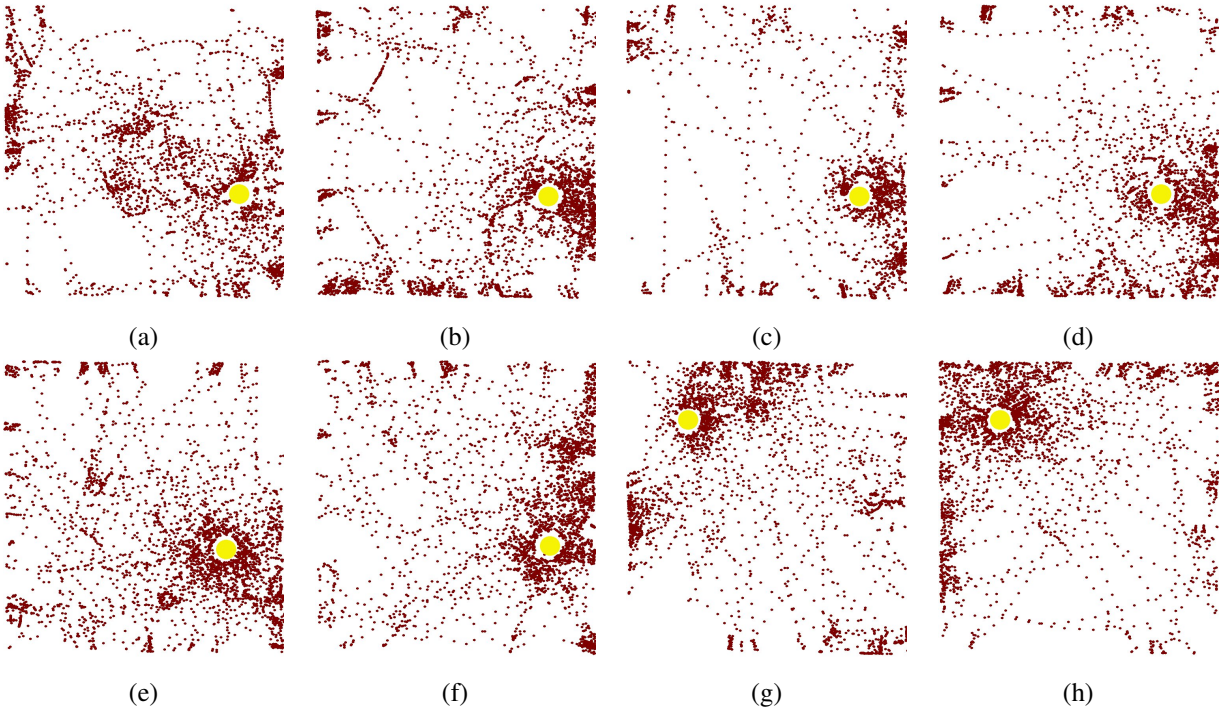

Figure 4: E-pucks' trajectories: each image records the locations of all e-pucks during 4 minutes, using a one location per e-puck per second resolution

be achieved with dedicated selection and/or variation operators such speciation is very (but not completely) unlikely to appear spontaneously. As with previous studies about the evolution of a homogeneous behaviour, the number of mating opportunities is a critical parameter as few opportunities are mandatory for the evolution of speciation.

Most experiments have been performed using the Roborobo simulator[3], originally developed by Nicolas Bredeche, which is a simulator for large population of robotic agents. It is programmed in C++ using the SDL graphics library. It features basic robotic-inspired agent dynamics with friction-based collision, and provide *very* fast simulation (> 1000 position update per second with 100 virtual robots resembling e-pucks). While most extensive tests are performed in simulation (due to time constraint limitation), the model under scrutiny is also considered valid with respect to real world implementation thanks to occasional real world validation, such as described in the previous section.

Note that *all* source code and parameters for all experiments published so far are available on-line in the Evolutionary Robotics Database[4].

[3] `http://code.google.com/p/roborobo`
[4] `http://www.isir.fr/evorob_db/`

Discussion and Related Works

A related research area is Embodied Evolution (Watson et al., 2002) (EE), a sub-field of Evolutionary Robotics that deals with the on-line, possibly distributed, evolutionary acquisition of behaviours. In most cases, EE considers a population of robots that interact with one another depending on spatial relations - and bares some similarities with Island Models in Evolutionary Computation (EC). As with other EC algorithms, EE is **objective-driven**, that is the user defines a particular objective function to optimize. In most works so far, these objective functions are used to optimize individual skills, with some very few exceptions targeting coordinated behaviours (Prieto et al., 2010).

Objective functions may be characterized in various ways, depending on the detail of description (explicit vs. implicit), type of description (behavioural vs. functional) or origin of information used to compute the fitness (internal vs. external) (Nolfi and Floreano (2000), p.67). At this point, it should be noted that the terminology is sometimes prone to confusion regarding the term *implicit fitness*. Indeed, there is a strong difference between objective-driven evolution and **environment-driven** evolution, which is what is considered throughout this paper.

On the one hand, objective-driven evolution requires the human user to write down a function used to evaluate the performance (or *utility*) of a particular behaviour. In this scope, an "implicit fitness" designates the fact that the user

Objective-driven *(a user-defined metric is used to compare individuals)*		**Environment-driven** *(the environment does the selection)*	
explicit fitness	*implicit fitness*	*biased natural selection*	*unbiased natural selection*
EE E.g.: maximize travelled distance, maximize #items harvested, ...	EE, ALife E.g.: maximize energy, novelty search, ...	ALife E.g.: AVIDA's metabolic function	ALife, EDEA E.g.: Tierra's natural selection

Figure 5: Classification of Embodied Evolution (EE), Open-ended Evolution in Artificial Life (ALife) and Environment-driven Distributed Evolutionary Adaptation (EDEA): a perspective from the nature of the selection pressure. Note that (1) this table is limited to contributions that deals with on-line evolution (ie. Evolutionary Robotics is omitted, but otherwise would match EE classification); (2) this table ignores other important axis that could be used to discriminate among these research fields, in particular wrt. the motivation (modeling the real world vs. engineering).

explictly designed a fitness function which description gives few indications on how to achieve an "efficient" behaviour. As an example, an *implicit fitness function* may be used to assign higher fitness values to a particular individual that maximizes its own energy level, while this could ultimately depend on displaying a particular skill such as the ability to aggregate with other robots (see (Weel et al., 2012) for an example). In other words: while an implicit fitness function gives loose indications on the behaviour to learn, it enables to *compare* candidate behaviours, therefore making it possible to perform non-uniform selection (as usual with evolutionary optimization).

On the other hand, environment-driven evolution implies that evolution is performed *without any fitness function description* - i.e. in its seminal form, that means than selection pressure occurs at the level of the population, but cannot be modelled at the level of the individual due to the inability to evaluate individual performance. In other words, random selection at the level of the individual is performed, but selection pressure will act at the level of the population, as with mEDEA, but also with various artificial life systems dealing with open-ended evolution, such as Tierra (where selection pressure results from memory limitation).

Figure 5 suggests a classificiation of the different fields discussed in this Section, from the perspective of the nature of the selection pressure considered. Of course, the limit is often unclear. Many open-ended evolution models that can be found both in Artificial Life (e.g. contributions using Genetic Algorithm to model the selection/replacement process) or Evolutionary Biology (Gavrilets, 2004) rely on the more explicit formulation of an implicit fitness, as it makes experimental studies more controllable and easier to interpret. Avida's metabolic function can also be seen as an indirect way to favor objective-driven selection (though it does not automaticaly imply reproduction). The limit is not clear either when it comes to biased vs. non-biased environment-driven selection. As an example, *assortative mating* stands as a mechanism that bias natural selection due to a particular mating strategy.

An additional concern would then be to evaluate if the definition of a fitness function results either from a pre-defined implementation choice from the user (*user-defined* objective-driven function) or from the evolution of phenotypic traits that can be used by individuals in order to select particular candidates for mating (*evolved* objective-driven function or mating strategy). As an example, a population of robots could learn an efficient cooperative behaviours to survive in a given environment, *then* learn to display their altruistic nature through the occurrence of a phenotypic trait such as growing a green beard (for example). Such a phenotypic trait could then be used as a basis to evolve a non-uniform selection mechanisms for mating and reproduction.

Open Problems with EDEA

The scientific question behind this work is to produce algorithms to learn efficient behaviours w.r.t. survivability in the context of open-ended environments, where selection pressure is solely driven by the environment. As such, the roadmap should follow an engineering viewpoint, listing and successfully addressing a list of challenges that may occur in the real world. In addition to those already described in the previous Section, it is possible to identify several issues that remain to be addressed. While the list to follow is far from exhaustive, we can already sketch three (complementary) research directions:

- **Environment-driven Evolution**: extending existing works (see previous Section), several issues may be addressed from the evolution of communication, of cooperation (division of labor), of developmental, epigenetic and learning mechanisms, to the evolution of phenotypic traits and mating/reproductive strategies, to name a few. For some of these issues, several other domains (such as Evolutionary Biology, Evolutionary Robotics, Swarm Robotics or Machine Learning) share similar concerns and can provide many insights and information w.r.t. our own objectives.

- **Defining an optimal performance zone**: a human en-

gineer may possibly define *a priori* a set of restrictions over possible actions to be taken, both at the individual or group levels. For example, one may implement restrictions with respect to what can be considered as dangerous actions (e.g. falling into a hole, breaking contact with other robots, ...). This can be achieved by particular choices of implementation, such as using a subsomption architecture, with an EDEA running as a meta-behaviour with low priority, and dedicated behaviours for critical situations. As an example, one may want to enforce particular communication graph topological properties, such that a robot cannot get lost voluntarily. The main issue is then to address the trade-off between restricting the so called performance zone, where any authorized actions can be performed, and relaxing such constraints so as not to limit behaviours that can be evolved.

- **Controlling the swarm**: in this work, an underlying hypothesis is that self-sustainability must be achieved before addressing user-controllable robot swarm. This usually implies the ability to recharge one's battery from time to time, but may also require more complex organization at the level of the population depending on the environment at hand (see above). Indeed, one important objective is to mix both environment-driven and user-defined objective-driven evolution so that a swarm could achieve one particular tasks without the user having to deal with robot's self-sustainability issues. Then, controllability can be interpreted in many ways, from endowing all robots with a global utility function (which may be reformulated within each robot) to on-the-fly control through interaction between the user and one (or several) robot(s), as well as specific robot recruitment strategies.

Beyond swarm robotics, EDEA may also be of interest whenever distributed processing among a large set of autonomous components is involved. Indeed this kind of situation is ubiquitous in todays environment : the Internet, Cloud Computing, e-mobility, etc. In this scope, Autonomic Computing (Murch, 2004) considers one particular major challenge: how to endow distributed systems where multiple units interact on a local basis with a self-managing capability (i.e. self-configuration, self-healing, self-optimization, self-protection) as the user (or system administrator) may not be able to manage such complex systems by him/herself.

Concluding remarks

This paper provided a short introduction to on-going research done in algorithm design for Environment-driven Distributed Evolutionary Adaptation (EDEA). A rather simple algorithm was described that can be used to generate complex evolutionary dynamics within a swarm of real robots. A description of open issues and a discussion of related works was provided, in order to clarify the particular class of problems addressed.

The underlying motivations behind EDEA are both *fundamental*, i.e. understanding which mechanisms are required to address particular environmental conditions, and *applied*, as the ultimate goal is to design algorithms that can actually be deployed and used in the real world.

Acknowledgments

This work was made possible by the European Union FET Proactive Initiative: Pervasive Adaptation funding the Symbrion project under grant agreement 216342. The authors also wish to thank several people with whom, at some point, we collaborated: Wenguo Liu, Alan Winfield, Simon Carrignon, Antoine Sylvain, Leo Cazenille.

References

Beyer, H.-G. and Schwefel, H.-P. (2002). Evolution strategies – A comprehensive introduction. *Natural Computing*, 1:3–52.

Bredeche, N. and Montanier, J.-M. (2010). Environment-driven Embodied Evolution in a Population of Autonomous Agents. In Springer, editor, *The 11th International Conference on Parallel Problem Solving From Nature (PPSN 2010)*, pages 290–299.

Bredeche, N., Montanier, J.-M., Wenguo, L., and Winfield, A. F. (2012). Environment-driven Distributed Evolutionary Adaptation in a Population of Autonomous Robotic Agents. *Mathematical and Computer Modelling of Dynamical Systems*, 18(1).

Colombano, S. P. and Shen, W.-M. (2006). Self-sustaining robotic systems. *Autonomous Robots*, 20:83–84.

Engwirda, A. E. (2006). Sustainable polymorphic colonial robotics and large scale off-world construction. *Autonomous Robots*, 20:137–148.

Gavrilets, S. (2004). *Fitness Landscapes and the Origin of Species*. Monographs in Population Biology. Princeton University Press.

Hardin, G. (1998). Extensions of the tragedy of the commons. *Science*, 280(5364):682–683.

Hauert, S., Zufferey, J.-C., and Floreano, D. (2009). Evolved swarming without positioning information: an application in aerial communication relay. *Autonomous Robots*, 26(1):21–32.

Jakobi, N., Husband, P., and Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In Moran, F., Moreno, A., Merelo, J., and Chancon, P., editors, *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life*, pages 704–720. Berlin: Springer Verlag.

Liu, W. and Winfield, A. F. (2011). Open-hardware e-puck linux extension board for experimental swarm robotics research. *Microprocessors and Microsystems*, 35(1):60–67.

Montanier, J.-M. and Bredeche, N. (2011). Surviving the tragedy of commons: Emergence of altruism in a population of evolving autonomous agents. In *Proceedings of the 11th European Conference on Artificial Life*.

Murch, R. (2004). *Autonomic Computing*. IBM Press.

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

Prieto, A., Becerra, J. A., Bellas, F., and Duro, R. J. (2010). Open-ended evolution as a means to self-organize heterogeneous multi-robot systems in real time. *Robotics and Autonomous Systems*, 58(12):1282–1291.

Ray, T. S. (1991). An approach to the synthesis of life. In Langton, C., Taylor, C., Farmer, J. D., and Rasmussen, S., editors, *Artificial Life II*, volume XI of *Santa Fe Institute. Studies in the Sciences of Complexity*, page 371408. Addison-Wesley, Redwood City, CA.

Shen, W.-M., Krivokon, M., Chiu, H., Everist, J., Rubenstein, M., and Venkatesh, J. (2006). Multimode locomotion via superbot reconfigurable robots. *Autonomous Robots*, 20:165–177.

Watson, R. A., Ficici, S. G., and Pollack, J. B. (2002). Embodied evolution: Distributing an evolutionary algorithm in a population of robots. *Robotics and Autonomous Systems*, 39(1):1–18.

Weel, B., Haasdijk, E., and Eiben, A. (2012). The emergence of multi-cellular robot organisms through on-line on-board evolution. In di Chio, C., editor, *EvoApplications 2012*, number 7248 in LNCS. Springer.

Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., and Chirikjian, G. S. (2007). Modular self-reconfigurable robot systems: challenges and opportunities for the future. *IEEE Robotics and Automation Magazine*.

Annex: the mEDEA algorithm

The following description is copied and adapted from (Bredeche and Montanier, 2010).

Algorithm 1 The MEDEA algorithm

```
genome.randomInitialize()
while forever do
    if genome.notEmpty() then
        agent.load(genome)
    end if
    for iteration = 0 to lifetime do
        if agent.isAlive() and genome.notEmpty() then
            agent.move()
            broadcast(genome)
        end if
    end for
    genome.empty()
    if genomeList.size > 0 then
        genome = applyVariation(select_random(genomeList))
        if agent.isAlive == false then
            agent.setAliveState(true)
        end if
    else
        agent.setAliveState(false)
    end if
    genomeList.empty()
end while
```

The mEDEA algorithm ("minimal EDEA") is described in table 1. This algorithm is implemented in each robot and describes how evolution is handled localy. This algorithm works along with a communication routine (not shown here), which purpose is to receive incoming genomes and store these in the list of imported genomes for later use (*genomeList*).

At a given moment, an agent is driven by a control architecture (e.g. an artificial neural network) which parameters are extracted from an "active" genome, which remains unchanged for a generation. This genome is continuously broadcasted to all agents within (a limited) communication range. This algorithm actually implements several simple, but crucial, features, that can be interpreted from the viewpoint of a traditional evolutionary algorithm structure:

Selection operator: the selection operator is limited to simple random sampling among the list of imported genomes, ie. no selection pressure *on a local individual basis*. However, cumulated local random selection ultimately favor the most widespread genomes *on a global population basis* as such genomes have greater probability to be randomly picked *on average*. In fact, the larger the population and mating opportunities, the more accurate the selection pressure at the level of the population.

Variation operator: the variation operator is assumed to be rather conservative to ensure a continuity during the course of evolution. Generating altered copies of a genome only make sense if there is some continuity in the genome lineage: if no variation is performed, the algorithm shall simply converge on average towards the best existing genome initially in the population. In the following, we assume a gaussian random mutation operator, inspired from Evolution Strategies Beyer and Schwefel (2002), which conservative behavior can be easily tuned through a σ parameter.

Replacement operator: lastly, replacement of the current active genome to control a given agent is performed by (1) local deletion of the active genome at the end of one generation and (2) randomly selecting a new active genome among the imported genome list (cf. selection operator). On a population level, this implies that surviving genomes are likely to be correlated with efficient mating strategies as a given genome may only survive through (altered) copies of itself in the long run.

The positive or negative impact of environmental variability on genome performance is smoothed by the very definition of the variation operator as newly created genomes are always more or less closely related to their parent. As a consequence, each genome results from a large number of parallel evaluations, both on the spatial scale as closely related copies sharing the same ancester may evaluated in a population, and on the temporal scale, as one genome is also strongly related to its ancestors. Hence, a single genome may get lucky once in a while, but it's highly unlikely that a "family" of closely related genomes manage to survive in the population if there are more efficient competitors.