Exploring New Horizons in Evolutionary Design of Robots

Stéphane Doncieux and Jean-Baptiste Mouret and Nicolas Bredeche

***Abstract*— This introduction paper to the 2009 IROS workshop "Exploring new horizons in Evolutionary Design of Robots" considers the field of Evolutionary Robotics (ER) from the perspective of its potential users: roboticists. The core hypothesis motivating this field of research will be discussed, as well as the potential use of ER in a robot design process. Three main aspects of ER will be presented: (a) ER as an automatic parameter tuning procedure, which is the most mature application and is used to solve real robotics problem, (b) evolutionary-aided design, which may benefit the designer as an efficient tool to build robotic systems and (c) automatic synthesis, which corresponds to the automatic design of a mechatronic device. Critical issues will also be presented as well as current trends and pespectives in ER.**

I. INTRODUCTION

The advent of genetic algorithms in the sixties, as a computational abstraction of Darwin's theory of evolution, promised to transfer the richness and efficiency of living organisms to artificial agents, such as robotic systems. This envisioned future inspired a whole field of research, now called *Evolutionary Robotics* (ER) [1], [2], [3], in which researchers create evolutionary algorithms to design robots, or some part of robots such as their "artificial brain". The long-term goal of this field is to obtain an automatic process able to design, and even build, an optimal robot given only the specification of a task; the main underlying hypothesis is that Darwin's theory of evolution is the best source of inspiration, in particular because Nature demonstrated its efficiency; the main hope is to obtain machines that fully and robustly exploit the non-linear dynamics offered by their structure and their environment without having to model them explicitly.

After almost twenty years of research, simple crawling robots have been automatically designed then manufactured [4]; neural networks have been evolved to allow wheeled robot to avoid obstacles then autonomously charge their battery[5]; neural networks have also been evolved to drive walking [6], [7] and flying [8], [9] robots, as well as self-organizing swarm of robots[10], [11].

These results demonstrate that it is *possible* to automatically design robots or parts of robots with evolutionary algorithms. However, most evolved robots or controllers are not yet competitive with human-designed solutions. What was seen as complex challenges for robotics twenty years ago (walking robots with many degrees of freedom, non-linear control, simple but emergent reactive behaviors, ...) has now been widely investigated in robotics and many efficient solutions have been proposed. At the same time, evolutionary robotics matured too and it may be time to reconsider its place with regards to the robotics field.

Consequent to this analysis, this paper tackles the question: how current evolutionary algorithms can be used in current robotics? After a short reminder of Evolutionary Algorithms (EA) (section (II)), we describe the conditions of EA applicability (section III), i.e. when ER should be taken into consideration. We then review the main techniques developed in the ER field by dividing them into mature techniques (section IV-A), current trends (section IV-B) and long-term research (section IV-C). Last, we discuss the current challenges of ER and the corresponding perspectives (section V).

S. Doncieux and J.-B. Mouret are with ISIR Pierre and Marie Curie University, CNRS Pyramide Tour 55 Boite courrier 173 4, place Jussieu 75252 Paris cedex 05 France stephane.doncieux@isir.upmc.fr, mouret@isir.fr

N. Bredeche is with TAO - Univ. Paris-Sud, INRIA, CNRS LRI, Bat. 490, 91405 Orsay, France nicolas.bredeche@lri.fr

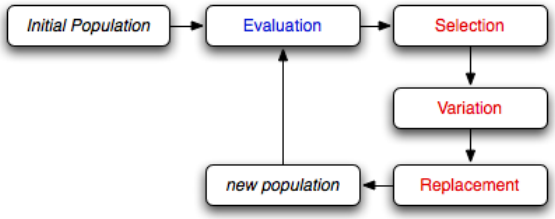

Fig. 1. Evolutionary Process: starting from a population of randomly generated individuals, each individual is evaluated. Based on the outcome of this evaluation, individuals are selected depending on their performance. Then, a new population is generated using various variation operators: mutation (ie. a new individual is created as a modified clone of a previous one) and recombination (ie. a new individual is created by merging several individuals of the previous generation. The evolutionary process goes on until a stopping criterion is reached.

II. A BRIEF INTRODUCTION TO EVOLUTIONARY COMPUTATION

Evolutionary methods are based on Darwin's principle of natural selection and blind variations [12]. The general scheme behind Evolutionary Algorithms is illustrated in figure 1. For more details on EA, interested readers may refer to [13], [14], [15]. EA share some principles with other meta-heuristics as they are iterative, stochastic and are targeted to provide (hopefully) good approximations in even the most extreme setup. EA also have their own particular features, as they are mostly population-based optimization algorithm, and heavily rely on blind variation over previously achieved solutions.

III. WHEN TO USE ER METHODS ?

Despite the large amount of papers about ER, the question of the underlying hypothesis of this approach is seldom discussed.

While there exists some active research providing sounded theoretical basis of Evolutionary Algorithm[16], the practical use of such methods does not require strong mathematical know-how so as to be efficient in any context. This section attempts to provide an overview of some critical aspects of using Evolutionary Algorithm in the context of Robotics.

A. Absence of "optimal" method

The first and foremost remark concerns the relevance of applying Evolutionary Algorithms rather than another existing methods to solve a given problem. Evolutionary Algorithms do not guarantee convergence towards a global optima, but merely provide an efficient way to address problems that are usually left aside because of their intrinsic difficulties (ill-defined, poorly-defined, implying complex dynamics, etc.). In this scope, ER results from a compromise between applying an iterative algorithm, that may be very slow compared to analytical method, and obtaining approximated solutions rather than no solution at all. Moreover, a key advantage of Evolutionary Robotics is its anytime nature, i.e. the ability to provide one or several solutions, more or less valid, whenever the algorithm is stopped.

B. Knowledge of fitness function primitives

EA principles consist in producing some diversity and then applying a selective pressure to, statistically, keep the best solutions and discard the others. The key question is that of defining what makes a solution better than the others? The behavior of solutions needs to be quantitatively described. To this end, descriptors of the behavior have to be defined and measured during an evaluation. Such descriptors are the fitness function primitives that should lead the search process towards interesting solutions.

There is no handbook to guide the design of such functions. It is often easy to define objectives able to discriminate between individuals that solve the problem – the preference going to those solving it faster or more efficiently – and likewise it is trivial to discriminate between individuals solving the task and those who don't solve it at all. The most difficult part of a fitness function design comes when individuals not solving the task at all have to be discriminated. For the algorithm to work, this discrimination should lead towards interesting solutions, but naive fitness functions often lead to local extrema, far from interesting solutions. Examples of such cases are numerous, the most famous probably being the obstacle avoidance problem. If simply defined as a count of collisions to be minimized, then the best way to minimize it is ... not to move at all ! Even if the robot is forced to move, it is simpler to find a way to turn round in a safe area, rather than to take the risk of coming close to obstacles and then of learning to use sensors.

C. Knowledge of phenotype primitives

The phenotype is the system to be designed by evolution. In Evolutionary Robotics, it may be a morphology, a control system or both. The goal is to find a design that best answers to the requirements quantitatively described within the fitness function. Evolutionary algorithms can do more than mere numerical optimization, it can also design complex structures like graphs (neural networks, for instance), set of rules, etc. Actually, EA may both assemble and parameterize sets of elementary or primitive elements. Solving a problem other than parameter optimization with an ER approach implies then to find appropriate phenotype primitives that will be assembled or modified by the genetic operators. For the search to be efficient (and at least more efficient than a pure random search), some important properties of the mapping between a genotype and a phenotype have been identified. Such properties have been studied in the Evolutionary Computation field and regrouped in three main categories, namely, redundancy, scaling and locality. For more details on these properties, see [17].

D. Appropriate genotype-fitness mapping

When considering ER experiments, there is another step between phenotype and fitness: the evaluation, which consists in a temporal sequence of strongly related events. An action of the robot may have a consequence in the environment that will then maybe trigger a different action of the robot, etc. How do the properties of locality, redundancy and scaling transfer to the genotype-fitness mapping? Such a question hasn't been studied yet, whereas it seems to be a critical one.

IV. WHERE AND HOW TO USE EA IN THE ROBOT DESIGN PROCESS ?

We will distinguish three different uses of EA in a robot design process:

- parameter tuning
- evolutionary aided design
- automatic synthesis

All of them do not have the same maturity. Parameter tuning consists (figure 2 (a) and (b)) in using EA as an optimization tool, this is their most frequent use, for which very efficient algorithms now exist, like CMA-ES[18], for instance, or NSGA-II for multi-objective problems [19]. Evolutionary aided design is a more recent trend that differs from parameter tuning in the use of the results. Whereas in parameter tuning, finding optimized parameters is the goal and generally comes at the end of the design process, in evolutionary-aided design, these optimized parameters are to be analyzed by experts to get a better understanding of the problem. Experts will then be able to propose new solutions[1] in a further step. Lastly, one promising use of EA is evolutionary synthesis. Evolutionary Synthesis is indeed the original motivation behind ER, i.e. building from scratch an autonomous agent by taking some inspiration from the

[1] whose parameters might be further tuned with an EA.

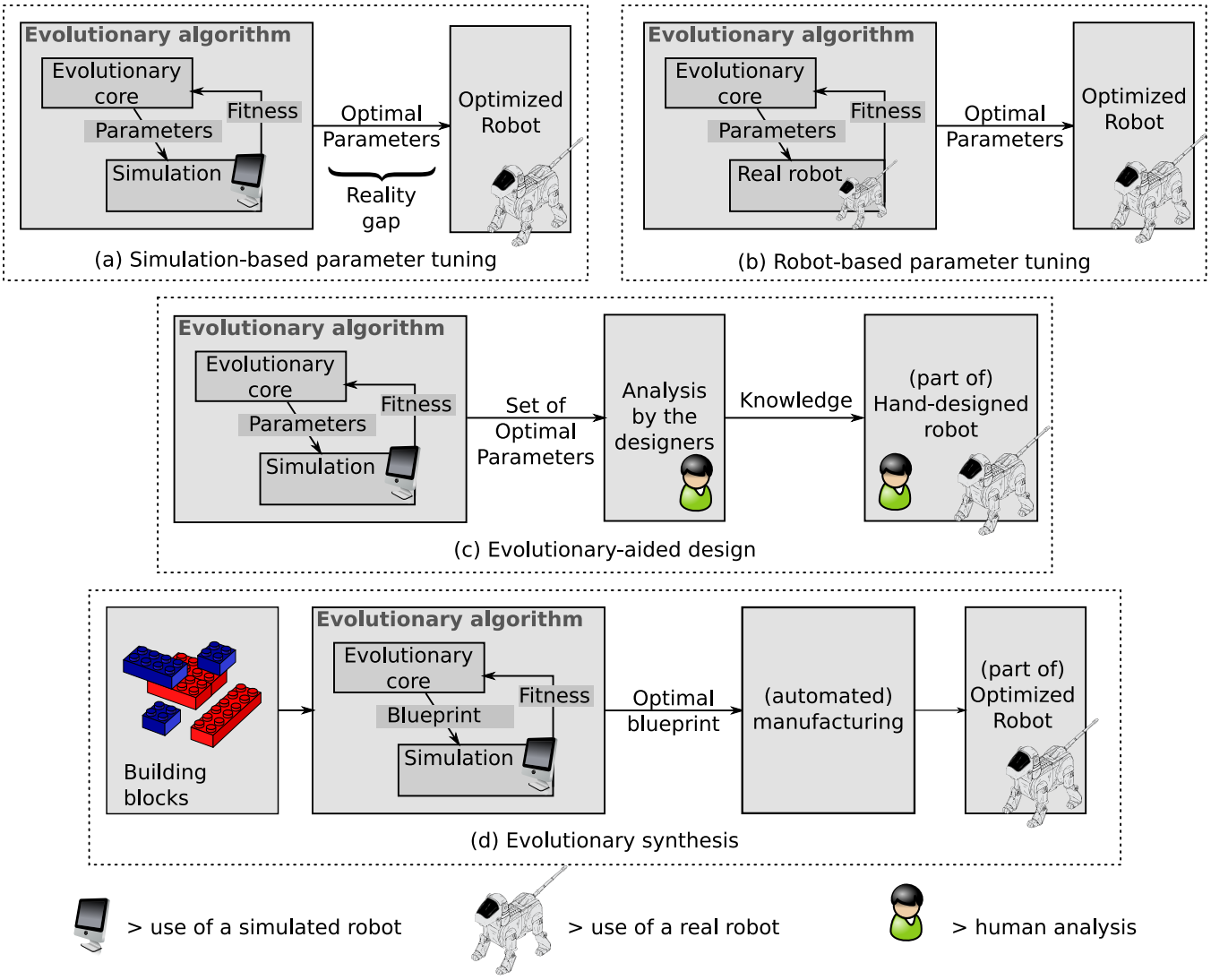

Fig. 2. Overview of the different uses of evolutionary algorithms in robotics. On this figure, "evolutionary core" denotes the basic evolutionary loop (see section II) excluding fitness evaluation. (a) Parameter tuning based on a simulation then a transfer to the real robots; (b) Parameter tuning that uses the real robot to evaluate the fitness; (c) Evolutionary-aided design (e.g innovization); (d) Evolutionary synthesis (building blocks can be neurons, physical blocks, ...).

actual evolution mechanisms. However, due to its challenging goal, it is also the less mature use of ER as many issues remain to be studied.

A. *Mature techniques: parameter tuning*

Evolutionary algorithms, and especially modern evolution strategies [20], are now mature tools for black-box optimization. As they don't impose any constraint on the objective function(s), they can be employed to tune some parameters (constants used in control laws, width of some parts, lengths, ...) of a robot with regards to a set of defined objectives. Typical applications work from a dozen to one hundred real parameters; they involve one to four objectives [21].

One of the easiest setup is to use a robot simulator combined with an EA to find the optimal parameters of a control law [22]. For instance, Kwok and Sheng [23] optimized the parameters of PID controllers for a 6-DOF robot arm with a genetic algorithm. The fitness function was the integral of sum of squared errors of joints, evaluated with a dynamic simulation of the robot. In addition to the many papers that propose to optimize classical control laws, a substantial litterature employed EAs to find optimal parameters of neural networks or fuzzy controllers (see [22] and [24] for some overviews), especially because such controllers are difficult to tune by hand.

Since simulators are never 100% realistic, results obtained in simulation often face what is called "the reality gap": the optimal parameters obtained in simulation may not be optimal on the real robot; in many cases, the optimized controller may even rely on so badly simulated behaviors that it does not work at all on the real robot. The potential solutions to bridge this reality gap will be described in section V-A.

B. Current trends: evolutionary aided design

A growing trend in evolutionary robotics is to use evolutionary algorithms for analysis and exploration tool instead of optimization. Hence, the main goal is not to find an optimal set of parameters but to answer questions such as:

- is it possible to solve problem X using the system parametrized by Y?
- what efficiency is to be expected if choice Z is made?
- How antagonistic objectives $O_1, ..., O_n$ are? Can we find a solution that is optimal with regards to all these objectives?
- does some regularities exist between optimal solutions?
- what are the critical parameters ?
- ...

The typical process is divided into three steps: (1) run an evolutionary algorithm (typically with a simulated system to evaluate the fitness); (2) analyze the results to have a better understanding of the studied system; (3) implement a solution on the real robot with classic (non-evolutionary) techniques but by exploiting the new knowledge to improve the design.

Such an approach was followed by Hauert et al. [25] to evolve decentralized controllers for swarms of unmanned aerial vehicles (UAV). They first evolved neural networks to automatically discover original and efficient strategies. In a second step, they reverse-engineered the obtained controllers to hand-design controllers which capture the simplicity and efficiency of evolved controllers. The hand-design step allows to check the generality of the controllers and to use well etablished methods – to guarantee the stability of controllers, for instance – while taking advantage of the potential innovations brought by the evolutionary process.

Deb and Srinivasan recently demonstrated how multiobjective evolutionary algorithms (see [21]) can bring knowledge of a given system through the analysis of Pareto-optimal solutions, a process called *innovization* [26], [27]. The first step consists in selecting two antagonistic objectives (or more); an evolutionary algorithm is then employed to find the best possible approximation of the Pareto Front; last, Pareto-optimal solutions are analyzed, for instance to find relations between parameters. Typical conclusions are:

- parameter X is constant for all the Pareto-optimal solutions;
- parameter X can be computed as a function of parameter Z;
- parameter X is critical;
- performance seems limited by the range of authorized values for parameter X;
- ...

This analysis can then be employed to reduce the number of parameters and/or to hand-design some efficient solutions. This approach has been successfully employed to design motors [26] and controllers of a flapping-wing robot [28].

C. Long term research: automatic synthesis

As Nature demonstrates it daily, Darwinian evolution is not solely an optimization tool, it is also a powerful automatic *design* process. The marvels accomplished by evolution inspired many researches with the long term goal of automatic designing and even manufacturing complete robotics "lifeforms" with as little human intervention as possible. From the robotics point of view, such an automatic design process could lead to "morpho-functional machines" [29], i.e. robots that can fully adapt the dynamics that emerge from the interactions between their morphology and their controller in order to optimally solve a task. The challenges raised by the automatic synthesis problems range from the understanding of biological evolution (what is the role of development to evolve complex shapes? how did living organisms evolved to modular systems?) to complex engineering problems (how could a robot be automatically manufactured, including its battery and its actuators?).

In a seminal paper, Sims [30] demonstrated how the morphology and neural systems of artificial creatures can be generated automatically with an evolutionary algorithm. Individuals were described as labeled directed graphs, which were then translated to morphology and artificial "brains". Sims was able to obtain creatures that managed to walk, swim and follow a target in a 3-dimensional dynamics simulator. The Golem project [4] put Sims' work in the robotics field by employing a 3D rapid prototyping machine to build walking robots whose morphology and controller were automatically designed by an evolutionary algorithm.

Despite these stimulating results, obtained creatures are by far many order of magnitudes simpler than any real organism. Many researchers hypothesized that designs have to be encoded using a representation that incorporates the principles of modularity (localization of functions), repetition (multiple use of the same sub-structure) and hierarchy (recursive composition of sub-structures) [31], three features of most biologically-designed systems but also of most engineered artifacts. Such principles led to several generative evolutionary processes that evolve programs that, once executed, generate a blueprint for a robot [32] or a neural network [33], [34]. Abstractions of the development process based on chemical gradients are also investigated [35], [36] and mostly employed to evolve neural networks. However, it has been found that these principles could need to be linked to appropriate selective pressures to be fully exploited [37], hence emphasizing that the synthesis problem may not be solely an encoding problem.

These experiments were all based on simulations, would it be to evolve neural networks to control robots or to evolve robotics morphology; some resulting structures have been successfully transferred to real robots. This avoids the complex problems of evaluating the fitness of robots whose morphology is evolved *on real hardware*. One of the idea to accomplish this task is to employ modular robots [38]. An alternative approach is to rely on rapid prototyping machines [4].

V. FRONTIERS OF ER AND PERSPECTIVES

ER still has many open issues. Here are several of the most critical:

- how to avoid the reality gap? Or, how to limit the risks of using an imperfect simulation to evaluate the performance of a system within an opportunistic learning scheme;
- how does it scale relative to behavior complexity? This question reveals to be actually tightly linked to fitness landscapes and exploration abilities of the EA. We will consider this question under this point of view;
- genericity of evolved solutions? For CPU time considerations, evaluations are as short as possible, and correspond thus to the behavior of the robot within only a limited set of conditions;
- can ER be applicable to address online learning?

We will briefly discuss them in this section and sketch out current work and perspectives.

A. Reality gap

The reality gap problem is clearly the most critical one with regards to practical applications. In theory, the reality gap should not even exist as the optimization process could be achieved directly on the target robotics setup. Several works have actually achieved evolution on real robots, such as for evolving homing behavior for a mobile robot [5], optimizing the walking gait of an AIBO robot [7], of a pneumatic hexapod robot with complex dynamics [39] or even a humanoid robot [40]. While the optimization on the real robot guarantee the relevance of the obtained solutions, this has several major drawbacks as it can be quite consuming term of time. As a consequence, only small populations (most of the time less than 30) and few generations (often less than 500) are performed in such a context, therefore limiting the problem that can be addressed to rather simple tasks.

Given that simulation is difficult to avoid in most practical situations, a new question arises regarding how to avoid, or at least limit, the reality gap effect, or, stated differently, how to ensure that the fitness function gives similar results within the simulation and on the robot. As a perfectly accurate simulation is highly unlikely to be available, many works focus on coping with the simulation intrinsic approximations and mistakes. A representive contribution is that of Jakobi [41] with minimal simulations: only the accurately simulated parts of the environment are taken into account and random noise is added to keep the evolutionary process from being mistakenly optimistic.

Another promising direction is to attempt to learn the desired features of the environment: instead of learning behaviors, ER techniques may be used to directly learn a model of a real mechanical device [42], [43], [44], [45]. Learning techniques can thus be used to correct model errors online [46] or even to learn a complete model of the robot in action [47], thus opening the way towards robots able to adapt to motor failures.

B. Fitness Landscape and Exploration

While Evolutionary Robotics has long been intended to address challenging problems, most of the achievements so far concern quite simply defined robotics problems: wall avoidance, food gathering, walking distance maximization, and other simple navigation tasks [48]. One of the major pitfalls is that the difficulty of a problem often arises with the complexity of the fitness landscape: while a smooth, convex fitness landscape with no noise will be quite easy to deal with, most of the problems from the real world often comes with multimodal, noisy fitness landscapes that feature neutrality regions. The direct consequence is that search may often get stalled, would it be at the very beginning of the algorithm execution (ie. a boostrap problem) or during the course of evolution (ie. premature convergence), with no hint on how to escape a local optimum or on how to direct the search within a region where all neighboring candidate solutions are equally rewarded.

Exploiting expert knowledge is a good way to escape from local optima, but as it is not always available, several solutions have been considered, the most prominent ones are listed here:

- decomposing the problem into sub-problems, each of them being solved separately, either implemented manually or learned. The resulting behaviors can then be combined through an action-selection mechanism, that may itself eventually be tuned through evolution; [49], [50], [51];
- reformulating the target objective into an incremental problem, where the problem is decomposed into possibly simpler fitness functions of gradually increasing difficulties, ultimately leading to what is reffered to as incremental evolution [52];
- reformulating the target objective into a set of fitnesses optimized independantly in a multi-objective context [53]. As opposed to the previous point, a multi-objective formulation of the problem makes it possible to avoid ranking sub-fitnesses difficulties, which is often a tricky issue;
- using co-evolution to build a dynamically changing evaluation difficulty in competitive tasks [54], [55];
- changing the evaluation during evolution to focus first on simpler problems and make the robot face progressively more difficult versions of the same task[56];
- likewise exploring solutions of increasing complexity with mechanisms protecting innovation to give new solutions a chance to prove their value[57]
- searching for novelty of behavior instead of efficiency [58]. This avoids getting trapped in local optima while enhancing the search ability over robot behaviors;
- in a multi-objective scheme, adding an objective that explicitely rewards the novelty or diversity of behaviors [59], [60], [61];
- putting the human into the loop. For instance, this is the kind of approach that have been previously called "innovization", where the search algorithm is used to provide a basis for the expert to refine the optimization process and to provide original solutions.

C. Genericity of evolved solutions

One major requirement of optimization in the context of ill or poorly defined problems is to provide solutions capable of generalization, or robustness. It may indeed be very difficult to grasp all the aspects of a problem during the conception phase as the combinatorial explosion makes it impossible to generate all possible test cases. A typical example is that of a walking robot where all inclinations or textures of the ground cannot be generated during optimization, but where generalization is possible over examples. In this setup, both the experimental setup and the representation formalism are of the utmost importance. For example, relying on a test case generator or adding noise during the course of evaluation is an efficient way to enforce generalization[62]. Also, some specific representations are more fitted for generalization: artificial neural network, for example, are naturally biased towards generalization whenever several close set of input values produce similar output values.

D. On-line ER: ER for on-line adapation

As stated earlier, Evolutionary Design tools for Robotics are considered as a specific flavor in the Optimization toolbox. Broadly, this means that Evolutionary Design is applied in an off-line manner, prior to the actual use in production of the obtained solution. While the solution at hand, would it be a relevant morphology and/or control architecture, may feature some kind of generalization capabilities, it is still limited to address a specific problem or class of problem, within a limited range of variability. On the other hand, Online Learning in Machine Learning addresses problem settings where the very definition of the problem is subject to change over time, either slowly or abrutly[63]. In this scope, the goal is to provide a continuously running algorithm providing adaptation in the long run, that is the conception and production phases happen simultaneously.

In the scope of ER, a sub-domain referred to as Embodied Evolutionary Robotics[64] focuses on this specific problem: an online onboard evolutionary algorithm is implemented into one robot or distributed over a population of robots, so as to provide real time adaptation in the environment. Advantages of this approach includes the ability to address a new class of problems (problems that require on-line learning), the parallelization of the adaptation (direct consequence of population-based search) and a natural way to address the reality gap (as design constraints enforce onboard algorithms). However this also comes with a price to pay: the lack of control over the experimental setup, such as the difficulty to reset the starting position of the robots inbetween evaluations, may dramatically slow down the optimization process. However, this field of research looks promising as it naturally addresses the unavailability of human intervention and/or environment control as the algorithm is supposed to be completely autonomous from the start. Indeed, a direct consequence is that most of the works in this context have been conducted on real robots[64], [65], [66], [67], which is sufficiently unusual in ER to be mentionned.

The long term goal of online ER is to provide continuous online adaptation by combining the ability to address the task specified by the human supervisor (the goal) with a priori unknown environmental constraints – that is constraints that cannot be expressed within the fitness function because of the a priori unpredictable nature of the environment. Hence, this field is at the crossroad of traditional optimization techniques, with an explicitly defined goal to address, and open-ended evolution, as the environment particularities are to be taken into account during the course of the adaptation process. Compared to other online learning techniques, evolutionary algorithms rely on the same advantages as for black-box optimization: the ability to provide robust optimization through stochastic operators in the scope of problems with limited expert's domain knowledge.

VI. CONCLUSION

After almost twenty years of research, ER has matured and offers new possible interactions with the robotics field. Many examples from the literature show that evolving robots structure or behavior is indeed possible. In the scope of this paper, three different domains of use have been identified: (a) parameter tuning (b) evolutionary aided design and (c) automatic synthesis. The most critical issues of ER have also been discussed, and the current trends as well as future perspectives have been highlighted.

REFERENCES

[1] S. Nolfi and D. Floreano, *Evolutionary Robotics: Biology, Intelligence and Technology of Self-organizing Machines.* The MIT Press, 2001.

[2] D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*, ser. Intelligent Robotics and Autonomous Agents. MIT Press, 2008.

[3] J. A. Meyer and A. Guillot, *Handbook of Robotics*. Springer-Verlag, 2008, ch. Biologically-inspired Robots.

[4] H. Lipson and J. B. Pollack, "Automatic design and manufacture of robotic life forms," *Nature*, vol. 406, no. 31, pp. 974–978, 2000.

[5] D. Floreano and F. Mondada, "Evolution of homing navigation in a real mobile robot," *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 1996.

[6] J. Kodjabachian and J. A. Meyer, "Evolution and development of neural networks controlling locomotion, gradient-following, and obstacle-avoidance in artificial insects," *IEEE Transactions on Neural Networks*, vol. 9, pp. 796–812, 1997.

[7] G. S. Hornby, S. Takamura, J. Yokono, O. Hanagata, T. Yamamoto, and M. Fujita, "Evolving robust gaits with aibo," in *IEEE International Conference on Robotics and Automation*, 2000, pp. 3040–3045.

[8] J.-B. Mouret, S. Doncieux, and J. A. Meyer, "Incremental evolution of target-following neuro-controllers for flapping-wing animats," in *From Animals to Animats: Proceedings of the 9th International Conference on the Simulation of Adaptive Behavior (SAB)*, S. Nolfi, G. Baldassare, R. Calabretta, J. C. Hallam, D. Marocco, J. A. Meyer, O. Miglino, and D. Parisi, Eds., Rome, Italy, 2006, pp. 606–618.

[9] Y. Shim and P. Husbands, "Feathered Flyer: Integrating Morphological Computation and Sensory Reflexes into a Physically Simulated Flapping-Wing Robot for Robust Flight Manoeuvre," *Lecture Notes in Computer Science*, vol. 4648, p. 756, 2007.

[10] R. Gross, M. Bonani, F. Mondada, and M. Dorigo, "Autonomous self-assembly in swarm-bots," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1115–1130, 2006.

[11] G. Baele, N. Bredeche, E. Haasdijk, S. Maere, N. Michiels, Y. Van˜de˜peer, T. Schmickl, C. Schwarzer, and R. Thenius, "Open-ended on-board evolutionary robotics for robot swarms," in *IEEE Congress on Evolutionary Computation, 2009 (CEC 2009)*, 2009.

[12] C. Darwin, *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*. London: John Murray, 1859.

[13] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing (Natural Computing Series)*. Springer, October 2008.

[14] *Evolutionary Computation 1: Basic Algorithms and Operators*, 1st ed. Taylor & Francis, May 2000.

[15] *Advanced Algorithms and Operations (Evolutionary Computation)*, 1st ed. Taylor & Francis, November 2000.

[16] D. Seminars, "Theory of evolutionary algorithms," 2000-2010.

[17] F. Rothlauf and Others, *Representations for Genetic And Evolutionary Algorithms*. Springer, 2006.

[18] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies." *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.

[19] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.

[20] N. Hansen, S. D. Muller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.

[21] K. Deb, *Multi-objectives optimization using evolutionnary algorithms*. Wiley, 2001.

[22] P. J. Fleming and R. C. Purshouse, "Evolutionary algorithms in control systems engineering: a survey," *Control Engineering Practice*, vol. 10, no. 11, pp. 1223–1241, 2002.

[23] D. P. Kwok and F. Sheng, "Genetic algorithm and simulated annealing for optimal robot arm PID control," in *IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*, 1994, pp. 707–713.

[24] D. Floreano and C. Mattiussi, *Bio-Inspired Artificial Intelligence: Theories, Methods, and Technologies*. The MIT Press, 2008.

[25] S. Hauert, J. C. Zufferey, and D. Floreano, "Reverse-engineering of Artificially Evolved Controllers for Swarms of Robots," in *IEEE Congress on Evolutionary Computation*, 2009.

[26] K. Deb and A. Srinivasan, "INNOVIZATION: Discovery of Innovative Design Principles Through Multiobjective Evolutionary Optimization," *Multiobjective Problem Solving from Nature: From Concepts to Applications*, p. 243, 2007.

[27] ——, "Innovization: Innovating design principles through optimization," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*. ACM New York, NY, USA, 2006, pp. 1629–1636.

[28] S. Doncieux, "Evolutionary algorithms as exploration and analysis helper tools, application to a flapping wing aircraft," in *Proceedings of IROS Workshop "Exploring New Horizons in the Evolutionary Design of Robots"*, 2009.

[29] F. Hara and R. Pfeifer, *Morpho-Functional Machines: The New Species: Designing Embodied Intelligence*. Springer, 2003.

[30] K. Sims, "Evolving virtual creatures," in *SIGGRAPH '94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM, 1994, pp. 15–22.

[31] H. Lipson, "Principles of Modularity, Regularity, and Hierarchy for Scalable Systems," *Genetic and Evolutionary Computation Conference (GECCO'04) Workshop on Modularity, regularity and Hierarchy*, 2004.

[32] G. S. Hornby, "Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design," *Proceedings of the 2005 conference on Genetic and evolutionary computation*, pp. 1729–1736, 2005.

[33] J.-B. Mouret and S. Doncieux, "Mennag: a modular, regular and hierarchical encoding for neural-networks based on attribute grammars," *Evolutionary Intelligence*, vol. 1, no. 3, 2008.

[34] F. Gruau, "Automatic definition of modular neural networks," *Adaptive Behaviour*, vol. 3, no. 2, pp. 151–183, 1995.

[35] J. J. Gauci and K. O. Stanley, "Generating large-scale neural networks through discovering geometric regularities," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007)*, 2007.

[36] D. B. D'Ambrosio and K. O. Stanley, "A novel generative encoding for exploiting neural network sensor and output geometry," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007)*, 2007.

[37] J.-B. Mouret and S. Doncieux, "Evolving modular neural-networks through exaptation," in *IEEE Congress on Evolutionary Computation, 2009 (CEC 2009)*, 2009.

[38] S. Kornienko, O. Kornienko, A. Nagarathinam, and P. Levi, "From real robot swarm to evolutionary multi-robot organism," in *2007 IEEE Congress on Evolutionary Computation*, D. Srinivasan and L. Wang, Eds., IEEE Computational Intelligence Society. Singapore: IEEE Press, 2007, pp. 1483–1490.

[39] H. Lipson, J. Bongard, V. Zykov, and E. Malone, "Evolutionary robotics for legged machines: from simulation to physical reality," *Intelligent Autonomous Systems 9*, p. 9, 2006.

[40] K. Wolff, D. Sandberg, and M. Wahde, "Evolutionary optimization of a bipedal gait in a physical robot," in *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.*, 2008, pp. 440–445.

[41] N. Jakobi, "Evolutionary robotics and the radical envelope-of-noise hypothesis," *Adaptive Behavior*, vol. 6, no. 2, pp. 325–368, September 1997.

[42] J. C. Bongard and H. Lipson, "Nonlinear system identification using coevolution of models and tests," *Evolutionary Computation, IEEE Transactions on*, vol. 9, no. 4, pp. 361–384, 2005.

[43] J. Bongard and H. Lipson, "Automated reverse engineering of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences*, vol. 104, no. 24, pp. 9943–9948, June 2007.

[44] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," *Science*, vol. 324, no. 5923, pp. 81–85, April 2009.

[45] S. Koos, J.-B. Mouret, and S. Doncieux, "Automatic system identification based on coevolution of models and tests," in *IEEE Congress on Evolutionary Computation, 2009 (CEC 2009)*, 2009.

[46] A. Gloye, F. Wiesel, O. Tenchio, and M. Simon, "Reinforcing the driving quality of soccer playing robots by anticipation (verbesserung der fahreigenschaften von fuballspielenden robotern durch antizipation)," *it - Information Technology*, vol. 47, no. 5/2005, pp. 250–257, May 2005.

[47] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, no. 5802, pp. 1118–1121, November 2006.

[48] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, 2001.

[49] N. Godzik, M. Schoenauer, and M. Sebag, "Evolving symbolic controllers," in *EvoWorkshops*, 2003, pp. 638–650.

[50] M. Wahde, "A method for behavioural organization for autonomous robots based on evolutionary optimization of utility functions," *Proceedings of the I MECH E Part I Journal of Systems & Control Engineering*, vol. 217, no. 4, pp. 249–258, 2003.

[51] K. J. Kim and S. B. Cho, "Robot Action Selection for Higher Behaviors with CAM-Brain Modules," in *Proceedings of the 32nd ISR (International Symposium on Robotics)*, vol. 19, 2001, p. 21.

[52] F. Gomez and R. Miikkulainen, "Incremental evolution of complex general behavior," *Adaptive Behavior*, vol. 5, no. 3-4, pp. 317–342, January 1997.

[53] J.-B. Mouret and S. Doncieux, "Incremental evolution of animats' behaviors as a multi-objective optimization," in *From Animals to Animats 10*, vol. 5040. Springer, 2008, LNCS, pp. 210–219.

[54] S. Nolfi and D. Floreano, "How co-evolution can enhance the adaptive power of artificial evolution: Implications for evolutionary robotics," in *Proceedings of the First European Workshop on Evolutionary Robotics (EvoRobot98*, 1998, pp. 22–38.

[55] K. O. Stanley and R. Miikkulainen, "Competitive Coevolution through Evolutionary Complexification," *Journal of Artificial Intelligence Research*, vol. 21, pp. 63–100, 2004.

[56] J. Auerbach and J. C. Bongard, "How Robot Morphology and Training Order Affect the Learning of Multiple Behaviors," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2009.

[57] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[58] J. Lehman and K. O. Stanley, "Exploiting Open-Endedness to Solve Problems Through the Search for Novelty," *Artificial Life*, vol. 11, p. 329, 2008.

[59] J.-B. Mouret, "Novelty-based multiobjectivization," in *Proceedings of IROS Workshop "Exploring New Horizons in the Evolutionary Design of Robots"*, 2009.

[60] J.-B. Mouret and S. Doncieux, "Using behavioral exploration objectives to solve deceptive problems in neuro-evolution," in *GECCO'09: Proceedings of the 11th annual conference on Genetic and evolutionary computation*. ACM, 2009.

[61] ——, "Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity," in *IEEE Congress on Evolutionary Computation, 2009 (CEC 2009)*, 2009.

[62] C. Hartland, N. Bredeche, and M. Sebag, "Memory-enhanced evolutionary robotics," in *IEEE Congress on Evolutionary Computation*, 2009.

[63] A. Blum, "On-line algorithms in machine learning," in *In Proceedings of the Workshop on On-Line Algorithms, Dagstuhl*. Springer, 1996, pp. 306–325.

[64] R. A. Watson, S. G. Ficici, and J. B. Pollack, "Embodied evolution: Embodying an evolutionary algorithm in a population of robots," in *1999 Congress on Evolutionary Computation*. Citeseer, 1999, pp. 335–342.

[65] ——, "Embodied evolution: Distributing an evolutionary algorithm in a population of robots," *Robotics and Autonomous Systems*, vol. 39, no. 1, pp. 1–18, 2002.

[66] Y. Usui and T. Arita, "Situated and embodied evolution in collective evolutionary robotics," in *Proc. of the 8th international symposium on artificial life and robotics*, 2003, pp. 212–215.

[67] J. M. Montanier and N. Bredeche, "Embedded evolutionary robotics: The (1+1)-restart-online adaptation algorithm," in *Proceedings of IROS Workshop "Exploring New Horizons in the Evolutionary Design of Robots"*, 2009.