

BACK TO REALITY: CROSSING THE REALITY GAP IN EVOLUTIONARY ROBOTICS

Juan Cristóbal Zagal, Javier Ruiz-del-Solar and Paul Vallejos

*Department of Electrical Engineering, Universidad de Chile
{jzagal, jruizd,pavallej}@ing.uchile.cl*

Abstract: In this work a new method to evolutionary robotics is proposed, it combines into a single framework, learning from reality and simulations. An illusory sub-system is incorporated as an integral part of an autonomous system. The adaptation of the illusory system results from minimizing differences of robot behavior evaluations in reality and in simulations. Behavior guides the illusory adaptation by sampling task-relevant instances of the world. Thus explicit calibration is not required. We remark two attributes of the presented methodology: (i) it is a promising approach for crossing the reality-gap among simulation and reality in evolutionary robotics, and (ii) it allows to generate automatically models and theories of the real robot environment expressed as simulations. We present validation experiments on locomotive behavior acquisition for legged robots.

Keywords: Evolutionary Robotics; Autonomous Systems; On-line Learning.

1. INTRODUCTION

Evolutionary robotics corresponds to an emerging paradigm for the automatic design and generation of robots, which are interpreted as autonomous artificial agents who are able to develop their skills and behaviors as the result from a tight interaction with their environment (Nolfy, Floreano, 2000). Within this framework the designer task consists on defining a fitness function for evaluating robot behaviors such that evolutionary search through solution spaces can be conducted. This paradigm has demonstrated to be successful in generating simple reactive robotic controllers, however there is much to be done in the design of complex robotic behaviors. Among the main attributes of this methodology we can mention that (Husbands and Harvey, 1992, Nolfi and Floreano, 2000) (i) it leaves away the designer bias in the generation of autonomous robotic systems, (ii) it allows machines to generate themselves into a theoretically open ended approach, and (iii) it allows the generation of robotic controllers, which are optimal to a given operational environment. A current fundamental limitation on real applications relies on the need of evaluating a large amount of robotic behaviors in their operational environment, with the corresponding waste of time and eventual platform damage. One alternative consists on simulating the interaction between the robot and the environment, which might be a complex and computationally expensive task. Simulation uses symbols and physical models to represent the real phenomenon that is perceived by the experimenter. Certainly this

introduces a strong bias when generating a simulator. No matter how accurate a simulator is, there is a substantial difference between simulation and reality, the so-called reality-gap. In the context of robotic behavior generation, crossing the reality-gap means that there is no difference between simulation and reality under the viewpoint of both, the robot controller and the external observer, who evaluates the behavior. This means that any behavior evaluation performed in reality gives the same outcome when performed in simulation.

In order to narrow the difference between reality and simulation Jakobi has proposed the minimal simulation approach (Jakobi, 1998), which consists on modeling just these aspects that are relevant for the controller development, making the strong assumption that the designer is able to identify *a priori* the relevant aspects that simulation should contain for the behavior evolution. We consider that this assumption is the main drawback of the minimal simulation approach.

In this paper we present a new method to evolutionary robotics which, by the inclusion of an illusory sub-system, allows crossing the reality-gap by generating representative simulators that are used by robots to quickly learn from their own experience. The remainder of the paper is organized as follows. In section 2 the organismic inspiration of this approach is exposed. In section 3 we discuss about the need of an illusory sub-system for producing complex robotic behaviors. In section 4 the theory of the proposed approach is presented. Section 5 presents validation experiments of this methodology. Finally, in section 6 some conclusions of this work are presented.

2. ORGANISMIC INSPIRATION

Organizational and working principles present in living organisms can be a source of inspiration for the development of autonomous robots. Animal behaviour provides a source of inspiration and aspirations for robotics. Animals have many of the functionalities that are missing in current robots, such as autonomy, intentionality and continuous learning. In the following concepts like autopoiesis, (M,R) systems, embodiment, auto-organization, emergence, structural coupling and homeostatic adaptation will be reviewed, and it will be analyzed how organismic principles can be employed in the development of autonomous robots.

The first issue to be analyzed is what defines a living system or putted in another way *what is life*. This question, formulated, among many others through history, by Schrödinger in his book “What is life?” (Schrödinger, 1944), has not been answered in a satisfactory way until the second half of the last century. Two theories, the (M,R) systems proposed by Rosen (Rosen 1991) and the autopoiesis set forth by Maturana and Varela (Maturana and Varela, 1980), have answered the question based on the same idea, the circular causality property of living systems, with an initial focus on the cellular metabolism. (Letelier have demonstrated the equivalence of both theories (Letelier *et al*, 2003)). According to the theory of autopoiesis (auto (self-) and poiesis (creation; production)) (Varela, 1979):

“An autopoietic system is organized (defined as a unity) as a network of processes of production (transformation and destruction) of components that produce the components that: (1) through their interactions and transformations continuously regenerate and realize the network of processes (relations) that produced them; and (2) constitute it (the machine) as a concrete unity in the space in which they [the components] exist by specifying the topological domain of its realization as such a network.”

Any unit meeting these specifications is an autopoietic system, and such autopoietic system realized in physical space is a living system. The particular configuration of a given unity, from now on its *structure*, is not sufficient to define a unit. The key feature if a living system is the maintenance of its *organization*, i.e. preservation of the relational network that defines it as a systemic unit. Putted in words of Maturana, “autopoietic systems operate as homeostatic systems that have their own organization as the critical fundamental variable that they actively maintain constant” (Maturana, 1975). The components of an organism are in constant flux due to its metabolism. As Di Paolo said, “organisms are a wave of matter and energy” (Di Paolo, 2003). However, organisms preserve their unity because their organization is maintained thanks to the circular causality property (stronger than auto-organization): a network of local transformations produces elements that maintain a boundary; at the same time this boundary captures the domain, in which the local transformations take place. In terms of the (M,R) systems theory, a metabolic

network (M) is composed by metabolic components and a repair subsystem (R) that produces (repairs) the metabolic components. Conversely, the components of the repair subsystem are produced by the metabolic network (Rosen, 1991).

A fundamental property of autopoietic systems is termed as *operational closure*. It means that an autopoietic system does not have any input or output, instead it creates a web of molecular processes that result in the maintenance of the autopoietic organization. Since an autopoietic system’s internal dynamics is self-determined, there is no need to refer any operational (or organizational) aspect to the outside. Thus the environment does not define the internal dynamics; it only perturbs the system’s dynamics. This does not mean that an autopoietic system is completely independent from its medium. It means that the system specifies its own internal states and the domain of its changes. In this context, external events act as perturbations that only trigger internal changes. But the magnitude and direction of these changes are defined by the internal dynamics of the system and not by the external perturbations (Maturana and Mpodozis, 2000). In this sense organisms are closed to efficient causes (Di Paolo, 2003).

The mechanism by which the autopoietic system and its environment determine, in a mutual way, some of their properties is called *structural coupling*. Structural coupling is the term for structure-determined engagement of a given unity with its environment or another unit. It is “... a historical process leading to the spatio-temporal coincidence between the changes of state” (Maturana, 1975). As such, structural coupling has connotations of both coordination and co-evolution. Structural coupling describes ongoing mutual co-adaptation without allusion to a transfer of some information across the boundaries of the engaged systems.

In the context of the theory of autopoiesis the term *autonomy* is defined as a systemic attribute, of which autopoiesis is a subset. An autonomous system (e.g. a robot) only needs to preserve its organization and not to produce its own components as an autopoietic system.

In the framework of autopoiesis *cognition* is contingent on embodiment, because this ability is a consequence of the structural coupling of the organism and its environment. From this perspective, the object of cognition (e.g. the environment) is necessarily qualified with respect to the (observing) organism. Cognition in the autopoietic view corresponds to the organism’s effective behavior within its domain of interactions. Thus, cognition is a matter of interacting in the manner in which one is able of interact, as a result of the experience (structural coupling). The process of cognition is embodied, not only in logical and inferential rules, but in the specific neurophysiological substrate with specific cognitive consequences where the nervous system can not distinguish illusion from perception (Letelier *et al*, 2003). A direct consequence of the embodiment of cognition is the fact that the cognition ability of an

autonomous system cannot be transferred from one to another. Cognition is defined by the structural coupling (the history of interactions) of each autonomous system and its environment.

Operational closure, structural coupling, as well as autopoiesis' concepts of organization, autonomy and cognition, seem to be organismic principles that should be employed in the development of autonomous robots.

The fact that the nervous system cannot distinguish illusion from perception, as all of us know from our experiences with dreams, flight simulators and video games, lead us to another idea: "To use illusions and real experiences for learning complex robot behaviors by means of a working paradigm that mimics the process of cognition in organisms and incorporates an illusory sub-system, the robot simulator; in this way the whole robot system, the robot controller and the robot simulator, is structurally coupled to the environment by continuous interactions with it".

We consider that simulation modeling, with its potential for capturing different levels of organization is a fundamental tool for designing autonomous robots. Structural coupling corresponds to a mutual property between an organism and its environment. We believe that an effective way of crossing the reality-gap is to structurally couple the robot controller-simulator with the environment during the learning process.

All these principles and ideas are included in the here proposed paradigm.

3. THE NEED FOR AN ILLUSORY SUB-SYSTEM

We consider that learning a concept is a problem more related with the complexity of the concept itself and the amount of training examples which are necessary for learning it, than with the learning method or algorithm being used. According to the *No Free Lunch Theorem* in the absence of prior information about the problem to be solved there are no reasons to prefer one learning algorithm or classifier model over another (Wolpert and Macready, 1995). If one learning algorithm seems to outperform another in a particular situation, it is a consequence of its fit to the situation and not to the general superiority of that algorithm. "When confronting a new pattern recognition problem, it is important to focus on the aspects that matter most – prior information, data distribution, amount of training data and cost or reward function" (Duda *et al*, 2001).

Increasing the amount of training examples given to a robot appears to be a fundamental point. One way of doing that is by just letting the robot to interact with its environment for a very long period. Another approach is to use a large amount of identical robotic platforms. Then, parallelization of task execution might be implemented with the strong assumption of behavior equality along platforms. This choice certainly contradicts the structural coupling and the principles of embodiment of cognition. Alternatively we consider that the more promising and expandable choice is to use an illusory sub-system, or simulator, for obtaining a large amount of training instances for behavior development. Thus instances can be obtained from both real as well as virtual experiences.

4. BACK TO REALITY

The *Back to Reality* (BTR) paradigm combines into a single framework learning in reality and learning in simulation. The main idea is that robot and simulator co-evolve in an integrated fashion as it can be seen in the block diagram presented in figure 1. The robot learns by alternating real and virtual experiences. Due to the embodiment of cognition, organisms cannot distinguish illusion from perception, in this case simulation from reality. This alternation work only when narrowing differences among simulation and reality; this is achieved by co-evolution of the simulator. Thus, the simulator is adapted from the robot interaction with the environment, by minimizing the difference in behavior fitness obtained in reality versus simulations.

The key issue to be stressed is that the robot system is defined by two components: the robot controller and the robot simulator. When the robot learns in reality, the robot controller is structurally coupled to the environment, when it learns in simulations the robot controller is structurally coupled with the simulator. Furthermore, considering the embodiment of cognition there is a structural coupling of the robot controller-simulator system (simulation world) with the robot controller-reality system (real world), see figure 1.

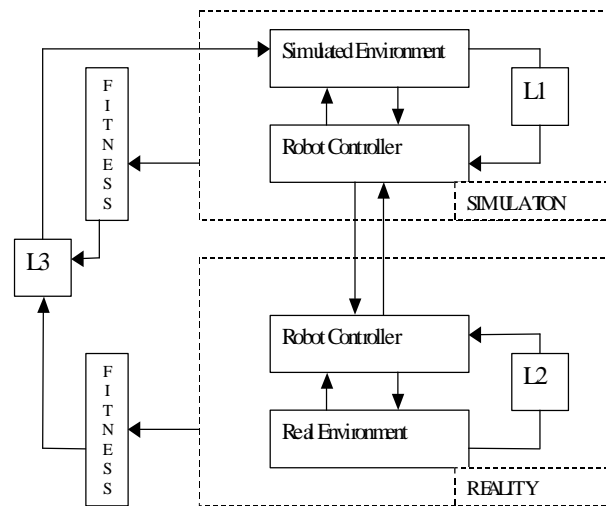


Fig. 1. Back to Reality building blocks.

In this way the simulation parameters are continuously tuned narrowing the reality-gap along the behavior adaptation process. Figure 2 shows how the reality-gap is narrowed by the BTR paradigm (one iteration of the procedure is shown). The first/second curve shows the fitness curve f_1/f_2 over the controller parameter space in reality/simulation; the maximal value of fitness f_1^*/f_2^* corresponds to the C_1/C_2 controller. The last graph illustrates how the difference between both curves is narrowed during the evolutionary process; the fitness curve of the simulator, defined by the simulator parameters, converges towards the (real) fitness curve of reality, using the difference between the obtained real and virtual fitness as adaptation parameter.

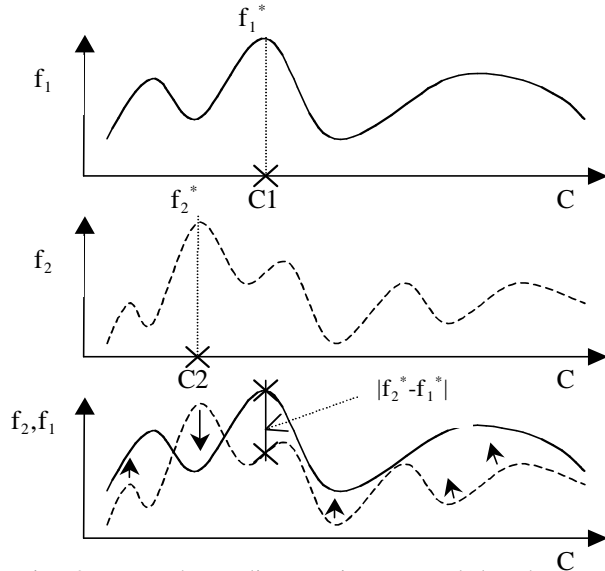


Fig. 2. How the reality-gap is narrowed by the BTR paradigm. See main text.

The BTR approach consists on the online execution of three sequential learning processes: $L1$ is the learning of the robot controller C in the simulated environment E' ; $L2$ is the learning of the robot controller C in the real environment E ; $L3$ is the learning of the simulator parameters performed by minimizing differences between the fitness obtained in reality and in simulation.

For implementing $L1$ and $L2$ any kind of learning algorithm can be used. Although we consider that given the evaluation time limitations of the experiments in reality, a reinforced learning algorithm is more suitable for implementing $L2$. Taking into account the flexibility of simulations, we believe a good alternative for implementing $L1$ are genetic algorithms. Regarding $L3$, we should consider the fact that the simulator has a large amount of parameters that probably are not explicitly related with aspects of the desired behavior. If this is the case, then $L3$ could be implemented using genetic algorithms. Otherwise, reinforced learning could be an alternative.

5. VALIDATION EXPERIMENTS

5.1. Learning to Walk:

We use the BTR approach in order to generate successful robot gaits. Our main motivation is to obtain better robot gaits in an automatic fashion. Since we have a team competing in RoboCup (Ruiz-del-Solar *et al*, 2003), we are particularly motivated with this subject given the strong correlation between the speed of robot-players and the success of a robot soccer team.

We will maximize the robot displacement speed measured in centimeters per second during evaluation trials of 20 seconds. The first stage of our experiment consists on using genetic search for exploring widely a solution space of the gait controller. In this stage we use a simulator as environment and a hand tuned solution as a starting point (although our approach well suites for starting from scratch). The second stage consists on evaluating in the real environment a set of successful behaviors obtained in the simulator and to

measure their corresponding fitness. These fitness measures are then used for optimizing the simulator parameters. The idea is that the simulator is continuously updated. A genetic algorithm searches trough the space of simulator parameters, and minimizes the difference between fitness values obtained in simulation and their values obtained in reality.

The third stage, which is executed in parallel, consists on learning in reality using policy gradient reinforcement learning. The idea is to perform smooth transitions away from the obtained solution using the reinforcement method. The resulting best solution which outcomes with reinforcement learning is then transferred back to the virtual environment were genetic search is again carried out. Then the final stage of the BTR cycle consists on testing in reality the resulting solution by going “back the reality”. This process can be repeated indefinitely.

For this experiment we use UCHILSIM, a realistic articulated rigid body dynamics simulator developed for the RoboCup four-legged league (Zagal and Ruiz-del-Solar, 2004).

5.1.1 Robot Controller Parameters

The gait locus used by our team is defined by a set of 20 parameters that give to our gait an extraordinary flexibility ranging from half-elliptical (Kohl and Stone, 2004) to trapezoidal loci (Röfer *et al*, 2003; Samut *et al*, 2003), passing throw a large number of combinations of them. By instructing each foot to move through a locus with each pair of diagonally opposite legs in phase with each other, and perfectly out of phase with the other two (a gait known as a trot), we enable the robot to walk. The gait implemented is composed basically by four stages: the foot on the ground, the foot lifting, the foot on the air and the foot descending. Nine parameters define the shape of the locus (length, shape factor and lift factor); the front locus shape modifier (lift height, air height and descending height); the rear locus shape modifier (lift height, air height and descending height). There are other seven parameters that define the operation point, these are the x , y and z coordinates of the front and rear feet, and an additional parameter which governs the AIBO turning rate that is used to determine the skew of all four loci in the x - y plane, a technique introduced by the UNSW team (Sammur *et al*, 2003). Other three parameters define the timing of the loci. (1) The speed of the foot while in the ground stage; (2) the lift time; and (3) the time on ground factor. Finally one parameter of processor charge is also used. It corresponds to the number of points of the air stage whose inverse kinematics is computed. Since the AIBO is roughly symmetric, the same parameters can be used to describe loci on both the left and right side of the body. To ensure a relatively straight gait, the length of the locus is the same for all four feet. Separate values for the front stage height, air stage height, rear stage height and position of the foot are used for the front and back legs.

5.1.2 Robot Simulator parameters

The robot simulator is defined by a set of 12 parameters, which determine the simulator and robot dynamics. These parameters include the values used for solving the dynamic equations and the PID constants used for modeling the leg joints. There are 4 parameters for the mass distribution in the robot: head mass, neck mass, body mass and leg mass; 4 parameters of the dynamic model: friction constant, gravity constant, and force dependent slip in two different friction directions; and finally 4 parameters for the joint leg model: proportional, integral and differential constants of the PID controller and maximum joint torque.

5.1.3 Learning to Walk

The learning to walk procedure consists on learning the 20 robot controller parameters in the simulator and in reality ($L1$ and $L2$), and learning the 12 simulator parameters ($L3$). Genetic algorithms are used for the evolution of the simulator parameters and for the evolution of the robot controller parameters in the simulator. Specifically, a conventional genetic algorithm employing fitness-proportional selection with linear scaling, no-elitism scheme, two-point crossover and mutation with a certain probability per bit was employed. Their corresponding settings are presented in section 5.3.

Given a set of controller parameters obtained from the simulator, we continued the adaptation of them in the reality using Policy Gradient Reinforcement Learning similarly as (Kohl and Stone, 2003). This method allows performing local estimation of the fitness function gradient by executing few behavior trials. Thus it is an attractive way for searching solutions locally in reality. The method starts from an initial parameter vector $\mathbf{p} = \{\mathbf{q}_1, \dots, \mathbf{q}_N\}$ obtained from the simulator (with $N=20$ in our case), and proceeds to estimate the partial derivative of \mathbf{p} 's objective function with respect to each parameter. We do so by first evaluating m randomly generated policies $\{R_1, \dots, R_m\}$ near \mathbf{p} , such that each $R_i = \{\mathbf{q}_1 + \Delta_1, \dots, \mathbf{q}_N + \Delta_N\}$ and each Δ_j is chosen randomly to be either $+\mathbf{e}_j$, 0, or $-\mathbf{e}_j$. Each \mathbf{e}_j is a fixed value small relative to \mathbf{q}_j . The evaluation of each policy generates a score that is a measure of the speed of the gait described by that policy.

5.2. Experiments Description

In order to obtain the fitness of a set of 20 controller parameters and 12 simulators parameters we evaluate the speed of the robot in simulation and reality. This is done in simulation by making the robot to walk during 20 seconds, and by calculating the total displacement at the end of the evaluation period. The displacement is then divided by the time, which is measured as 8ms per frame. A frame is the basic time unit of our actuation module. For evaluating the speed of the robot in reality, the robot is set in the center of a circle with radius 200cm, and then manually started and stopped through

button commands when it reach the circle border. During the real experiments, the robot measures the elapsed time using its internal clock assuming a travelled distance of 200cm, so the speed is calculated as the factor between the distance (200cm) and the elapsed time. The resulting speed is averaged over three trials.

The evolution of the walking parameters was done with a population of 15 individuals along 600 generations. The evolution of the simulator parameters was performed with 20 individuals along 45 generations. The learning in the real robots was done using 13 variations of the policy until the algorithm converges to an acceptable optimum.

5.3 Experiments Analysis

Figure 3 (top left) shows the evolution of maximal and average fitness over 600 generations of genetic adaptation of the robot controller on the simulator. An initial population was generated with mutations of the best hand tuned controller that our team had before this experiment (a controller which achieved 10 cm/seg). The graph shows results for populations of 15 individuals with mutation probability $Pm = 0.005$ per bit. It can be seen a clear improvement of fitness along generations. We should notice that some of the resulting best individuals obtained in simulation does not perform well in reality. There is a point in the adaptation process where the genetic algorithm exploits peculiarities of the simulator which do not exhibit a counterpart in reality. However translation to reality of about 60% of the best individuals of the evolved populations performed similarly in reality. We selected a set of 10 individuals exhibiting similar fitness in reality and simulation; they averaged a speed of 18 cm/s. The best individual of this group achieved 20 cm/s in reality. The fitness value of each one of these individuals was compared with the resulting fitnesses which they exhibit in simulation. And the norm of the resulting fitness differences vector was used as a fitness function to be minimized by genetic search through the space of simulator parameters. Figure 3 (top right) shows results of this minimization process. We obtained a minimum fitness of 2 cm/s as discrepancies among simulation and reality exhibited by these individuals. The best individual was then taken as a starting point for a policy gradient learning process performed in reality. With this method we achieved a speed of 23.5 cm/s. Figure 3 (bottom left) shows the evolution of fitness through this reinforcement learning process. The resulting best individual obtained with reinforcement learning was then taken back to the now adapted simulator and a genetic search took place starting with a population generated with permutations of this best individual. Figure 3 (bottom right) shows the evolution of controller fitness performed in the simulator.

Finally some of the best individuals resulting from the genetic adaptation were tested on reality. Among these trials we found an individual which averaged a real speed of 24.7 cm/s.

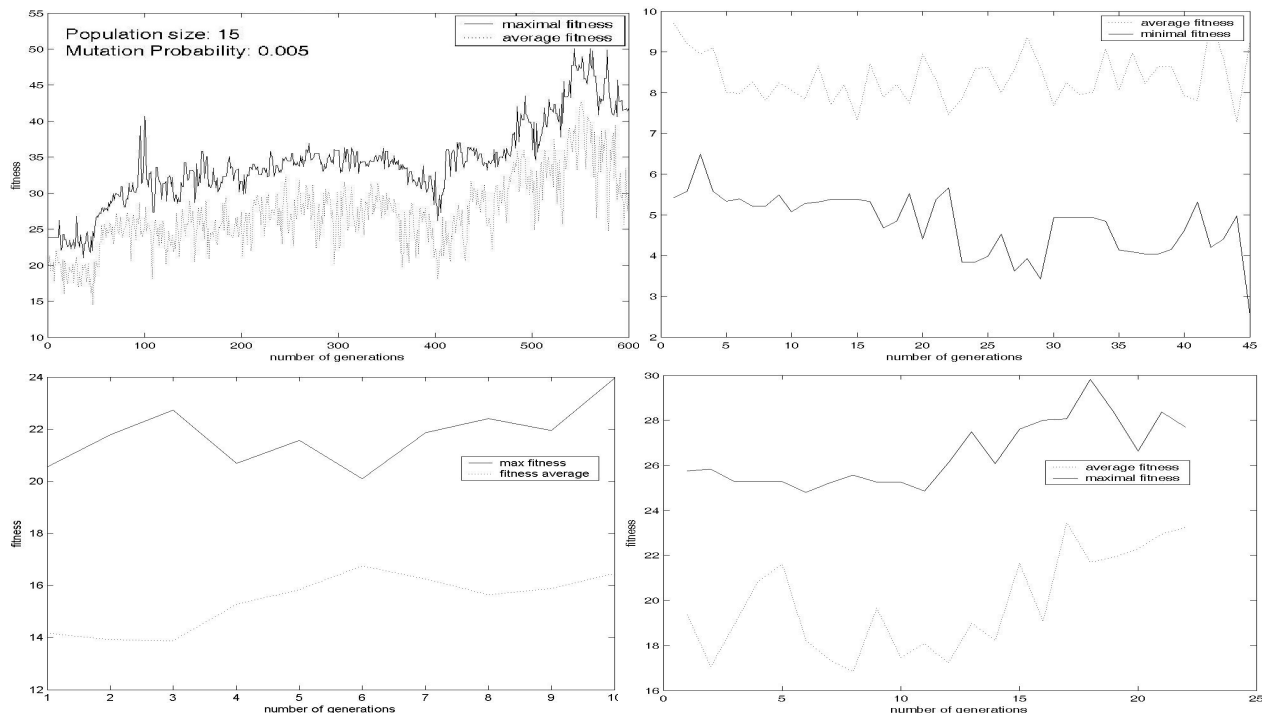


Fig. 3. Top left: Evolution of fitness for individuals tested in the UCHILSIM simulator. Top right: Minimization of discrepancies among simulator and reality through evolutionary search on the simulator parameter space. Bottom left: Evolution of velocity for gaits adapted with the policy gradient reinforcement method. Bottom right: Evolution of velocity for gaits adapted in simulation when going back to reality.

6 CONCLUSIONS

It took more than six months for our team to develop the robot gait of 10 cm/s. From this point, using BTR, we reached in just one week of work the speed of 24.7 cm/s. We consider that this is a remarkable improvement for our robot soccer team. It should be noticed that improvements were done for our own controlling system and therefore the resulting speed is not directly comparable to those obtained by others. Besides the gait locus used by a controller it also matters the efficiency on computations, the low level refinements on motor control, and the inverse kinematics models being used, etc. Given these results and those obtained in the recent work on learning to kick the ball with BTR (Zagal and Ruiz-del-Solar, 2004), we consider that the presented BTR paradigm is a promising tool for evolutionary robotics.

ACKNOWLEDGEMENTS

This research was funded by the FONDECYT Project 1030500 (CONICYT, Chile).

REFERENCES

Di Paolo, E. A., (2003). Organismically-inspired robotics: Homeostatic adaptation and natural teleology beyond the closed sensorimotor loop. In: *Dynamical Systems Approach to Embodiment and Sociality* (K. Murase & T. Asakura (Eds)), 19 – 42. Advanced Knowledge International, Adelaide, Australia.

Duda, R.O., Hart, P.E. and Stork, D.G. (2001), *Pattern Classification*, Second Edition.

Husbands, P., Harvey, I. (1992). Evolution Versus Design: Controlling Autonomous Robots. In: *Integrating Perception, Planning and Action: Proceedings of the Third Annual Conference on Artificial Intelligence, Simulation and Planning*, 139-146. IEEE Press.

Jakobi, N. (1998) Minimal Simulations for Evolutionary Robotics. PhD thesis, University of Sussex.

Kohl, N. and Stone, P. (2004). Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion. Submitted to ICRA 2004.

Letelier, J.C., Marin, G. and Mpodozis, J. (2003) Autopoietic and (M,R) systems, *Journal of Theoretical Biology*, **222**, 261–272.

Maturana, H. (1975) The organization of the living: A theory of the living organization, *Int. Journal of Man-Machine Studies*, **7**, 313 – 332.

Maturana, H., Varela, F. (1980). *Autopoiesis and Cognition: The Realization of the Living*. Reidel, Dordrecht.

Maturana, H., Mpodozis, J., 2000. The origin of species by means of natural drift. *Rev. Chil. Hist. Nat.* **73**, 261–310.

Nolfi, S., Floreano, D. (2000): Evolutionary Robotics – The Biology, Intelligence, and Technology of Self-Organizing Machines. In: *Intelligent Robotics and Automation Agents*. MIT Press.

Rosen, R., (1991). *Life Itself*. Columbia University Press, New York.

Ruiz-del-Solar, J., Zagal, J.C., Guerrero, P., Vallejos, P., Middleton, C. and Olivares, X. (2004): UChile1 Team Description Paper. In: *7th International Workshop on RoboCup 2003, Lecture Notes in Artificial Intelligence*, Springer, Padova, Italy.

Röfer, T., Dahm, I., Düffert, U., Hoffmann, J., Jüngel, M., Kallnik, M., Löttsch, M., Risler, M., Stelzer, M., Ziegler, J. (2004). GermanTeam 2003. *Proc. of the RoboCup 2003 Symposium*, Springer, Padova, Italy.

Sammut, C., Uther, W. and Hengst, B. (2003). A Description of the rUNSWift 2003 Legged Robot Soccer Team. *Proc. of the RoboCup 2003 Symposium*, Springer, Padova, Italy.

Schrödinger, E. (1944) *What is life?*, Cambridge Press, Cambridge.

Varela, F.J. (1979) *Principles of Biological Autonomy*, Elsevier, New York: (North Holland).

Wolpert, D. and Macready, W. (1995) No Free Lunch Theorems for Search, Tech. Report SFI-TR-95-02-010, Santa Fe, NM.

Zagal, J.C., Ruiz-del-Solar, J. (2004): UCHILSIM: A Dynamically and Visually Realistic Simulator for the RoboCup Four Legged League. In: *8th International Workshop on RoboCup 2004, Lecture Notes in Artificial Intelligence*, Springer, Lisbon, Portugal.

Zagal, J.C., Ruiz-del-Solar, J. (2004): Learning to Kick the Ball Using Back to Reality. In: *8th International Workshop on RoboCup 2004, Lecture Notes in Artificial Intelligence*, Springer, Lisbon, Portugal.