From Simulated to Real Robots.

Henrik Hautop Lund* and **Orazio Miglino****

*Department of Computer Science, Aarhus University,
Ny Munkegade, 8000 Aarhus C., Denmark
Tel.: +45 8942 3188 Fax: +45 8942 3255 E-mail: hhl@daimi.aau.dk

**Cognitve Science Group, Department of Psychology, University of Palermo and
Institute of Psychology, National Research Council, Rome, Italy
orazio@caio.irmkant.rm.cnr.it

Abstract - Evolutionary Robotics with the use of genetic algorithms to evolve control systems for real robots is a powerful tool, since it allows an automatic evolution of control systems. Yet, Evolutionary Robotics meets serious limitations because of the time consumption. It is very time consuming to evolve whole populations of real robots for many generations. A simulated/physical approach where main parts of the evolution takes place in a simulator reduces the time consumption dramatically. We describe how the Khepera miniature mobile robot can be used to build its own simulator with a semi-autonomous process, how to evolve neural network control systems for the Khepera robot in the robot's own simulator, and how to transfer the neural network control systems from the simulated to the real environment. By using this kind of simulator an expected gap in performance when transferring a robot control system from the simulator to the real environment is avoided.

Introduction.

In recent years much robotics research has focused on autonomous robots, and especially autonomous vehicles. Like natural organisms, autonomous robots collect informations from the external environment and produce motor actions that allow the robots to change their relations with the environment. The autonomy consists in having the robot itself to decide its actions based on the sensory input by providing the robot with its own onboard computer (microchip) and batteries. The use of Evolutionary Computation techniques to evolve control systems gives another dimension to the autonomy of the robots: it is not the researcher who implicitly design all possible behaviors of the robot. The behaviors of the autonomous robots emerge.

The autonomous robots approach is an interdisciplinary field that includes knowledge from engineering, computer science, psychology, and biology. Autonomous robots applications can be found in at least three areas: industrial prototypes, ethological and biological models, and educational programs. Industrial prototypes that can explore dangerous or varying environments are constructed and used for minefield exploration, lawn moving, cleaning machines, etc. Ethologists, biologists and psychologists build autonomous robots to replicate in more controllable settings (environments) behaviors observed in natural organisms (e.g., Deneubourg et al., 1992). Finally, teachers and educational psychologists use autonomous robots to facilitate the learning process about concepts regarding natural systems (e.g., Miglino and Lund, 1995; Pagliarini et al., 1995). What connects all these different applications is the work to construct a standard methodology to build control systems for real, autonomous robots. There has been various proposals on how to obtain this unifying goal of a standard methodology. The different proposals have been tested using mainly the Khepera miniature mobile robot (Mondada et al., 1994), or different constructions of LEGO robots. Our research work has included both tests on Khepera and LEGO robots, but here we will limit ourselves to describe some of the experiments with the Khepera robot only.

The Khepera miniature mobile robot (Fig. 1) has a circular shape (55 mm. of diameter, 30 mm. of height and 70 g. of weight). It is supported by two wheels and two small teflon balls. The wheels are controlled by two DC motors with an incremental encoder (10 pulses per mm. of advancement of the robot), and can move in both directions. The robot is provided with eight infra-red proximity sensors. Six sensors are positioned on the front of the robot, the remaining two on the back. A Motorola 68331 controller with 256 Kbytes RAM and 512 Kbytes ROM manages all the input-output routines and can communicate via a serial port with a host computer. By attaching Khepera to the host by means of a lightweight aerial cable and specially designed rotating contacts, one allows a full track and record of all important variables by exploiting the storage capabilities of the host computer; and at the same time it provides electrical power without using time-consuming homing algorithms or large heavy-duty batteries.

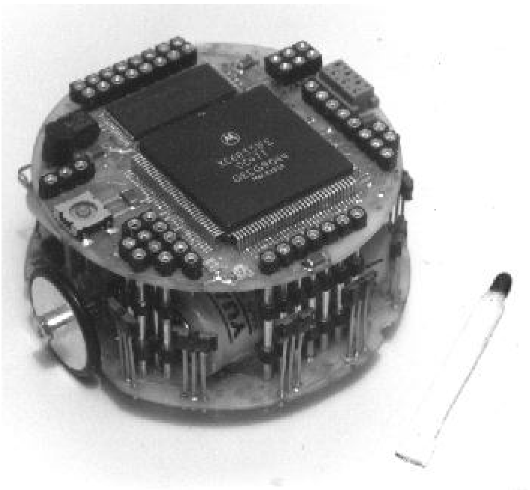

Figure 1. The Khepera miniature robot.

Model.

The control systems of autonomous robots can be chosen to be *Neural Networks*. Neural Networks are appropriate because of their adaptiveness, which is an important issue for autonomous robots. The adaptiveness allows the robots to work under different conditions in different environments. Another important character of Neural Networks is that learning algorithms do not require that the user manipulates the internal structure. This can be done completely by the learning algorithm. The user has just to present examples to the Neural Network, from which it can be trained by trial-and-error. A Neural Network control system for the Khepera robot can be a simple 2-layer feed-forward network with 8 input units that are connected to the 8 infra-red sensors, 2 bias units, and 2 output units that are connected to the 2 motors (see Fig. 2).

As an efficient learning algorithm for training Neural Networks used as control systems for autonomous robots we use a *Genetic Algorithm*. Initially, the Genetic Algorithm produces a population of Neural Networks (100 in our experiment) with randomly chosen connection weights. The Genetic Algorithm then evaluates each member of the population according to some fitness measure decided by the user, depending on the task to be solved. A subset (20) of the members that do best on the fitness measure is selected to reproduce. Each selected individual produces a number of offspring (5) by copying the individual Neural Network and introducing biological operators such as mutation and crossover. The testing and selective reproduction process is repeated until a satisfactory result is obtained.

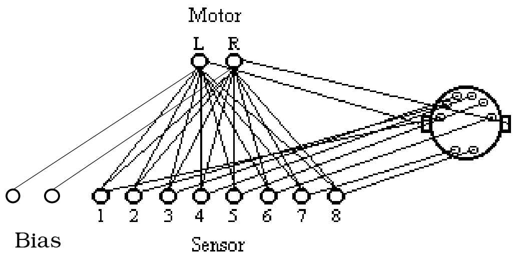

Figure 2. Connections between robot and neural network.

Working on populations of robots is a very time expensive process, since it involves the evaluation of each robot in the population over many generations (like is known from the natural evolution process that has been going on for millions of years). Therefore, as suggested in (Miglino et al., 1995), an appealing idea is to develop autonomous robots control systems in simulators before transferring the control systems to the real robots that act in the real environments. This, however, leads to the problem of being able to transfer control systems and structures perfectly from simulation to reality.

Since it is very difficulty to capture the fuzzy characteristics of the real world in a mathematical model, we build a simulator by letting the physical robot itself register its sensory activation in different parts of the surrounding environment and register the effect of different motor settings in the real environment. To illustrate this, let us look at an obstacle avoidance experiment for the Khepera robot. The desired behavior of the Khepera robot is based on the following abilities: (a) moving forward as fast as possible, (b) moving in as straight a line as possible, and (c) keeping as far away from objects as possible. In order to evaluate individual fitness in the Genetic Algorithm we used equation (1)

$$F = \sum_{i=1}^{steps} V_i (1 - DV_i^2)(1 - I_i) \qquad (1)$$

where V_i is the average rotation speed of the two wheels, DV_i is the algebraic difference between signed speed values of the wheels, and I_i is the activation values of the proximity sensor with the highest activity at time *i*. This value is summed over number of steps of the robot, *steps*.

We constructed the real, physical environment as a 60*35 cm rectangular arena surrounded by walls with 3 round obstacles of 5.5 cm placed in the center. Walls and obstacles were covered with white paper. In order to construct a model of this environment, Khepera empirically sampled the different classes of objects in the environment (wall and obstacles) through its own real sensors. It turned 360 degrees at different distances with respect to a wall and to an obstacle, while, in the meantime, recording the activation level of the sensors. The resulting matrices were then used by the simulator to set the activation levels of the simulated robot depending on its current position in the simulated environment. In the same way, to model the robot's motors, the effect of the different motors settings in the real world was sampled. How the robot moved and turned was modeled for all possible states of the motors. The obtained measures were used by the simulator to set the activation level of the neural network input units, and to compute the displacement of the robot in the simulated environment.

The procedure of letting the robot itself construct the model to be used by the simulator has several advantages: it is simple and it accounts for the idiosyncrasies of each individual sensor. It allows the building of a model of an individual robot taking into account the specificities of that robot that makes it different from other apparently identical robots. It also accounts for the idiosyncrasies of the environment by empirically modeling the environment itself, instead of building a mathematical model of it, as has been suggested by other researchers (e.g., Jakobi et al., 1995).

Results.

The time reduction is dramatically when using a simulator before transferring the evolved control systems to the real robot instead of making the whole evolution directly on the real robot by subsequently testing the different control systems on the same robot. In fact, a 98 % reduction in time consumption is obtained (It takes less than 1 minute to evolve a generation in a simulator that runs on an IBM RISC, while it takes at least 1 hour in the real world).

We construct the obstacle avoidance experiment in such a way that 200 generations are evolved in the simulated environment before the evolved Neural Network control systems are transferred to the Khepera robot. After this, the evolution continued for 20 generations in the real physical robot and environment.

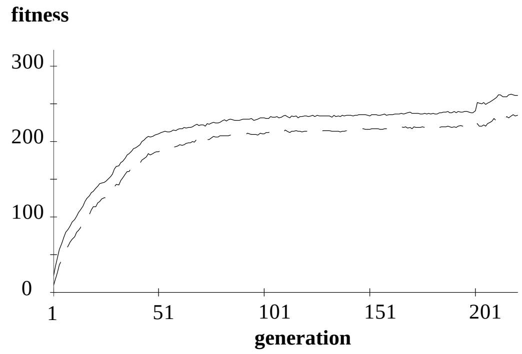

Figure 3. Peak and average performances as an average result of 5 experiments with noise. The first 200 generations are evolved in the simulated environment, the last 20 generations in the real environment.

Figure 3 shows the performance of the best individual, and the mean performance of each generation as an average of 5 experiments. The fitness increases quickly to a high level and already at generation 60 an optimal strategy seems to be found, and from this point on only minor increases in fitness will take place. Yet, the most remarkable result is observed when we transfer the control systems evolved in the simulator to the real Khepera robot at generation 200. There will be no decrease in fitness at all, and during the subsequent 20 generations we note an increase in fitness.

Also at the level of behavior, the evolved robots perform equally in the simulated and the real environment. If we take one of the individual of generation 200 and look at its behavior in the simulated and the real, physical environment, we observe that the trajectories of the mobile robots match almost perfectly, as shown in Fig. 4.

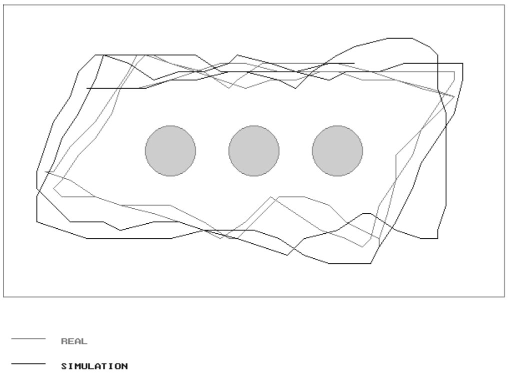

Figure 4. Robot behavior in simulated and real environment. Trajectory of the best individual of generation 200 of an experiment in the simulated (dashed line) and the real (full line) environment.

Conclusions.

The idea of Evolutionary Robotics that uses Evolutionary Computation techniques to evolve control systems for real robots is very appealing and with the results presented here, it has been shown how the main problem of time consumption can be avoided. It demands to build a reliable simulator of the robot and the robot's environment. This can be done by letting the robot itself register the characteristics of the environment and its own physics. We have shown, that with the use of such a simulator, we obtain a 98 % reduction in time consumption, and that there is a perfect match between the simulated and the real robots, both at the level of fitness and at the level of behavior.

The validity of using simulators opens new perspectives for Evolutionary Robotics. In simulators, it is possible to use fitness formulae, that can not be used on real robots, like distance to a given path, number of different cells passed, etc. Future work in Evolutionary Robotics should investigate how to evolve behaviors in simulators that can not be constructed without a simulator, and then transfer the behaviors to real robots in the real environment with the technique described in this paper.

References.

Deneubourg, J. L., Theraulax, G., and Beckers, R. (1992) Swarm-made Architectures. In F. Varela and P. Bourgine (eds.) *Toward a Practice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life.* MIT-Press, Cambridge.

Jakobi, N., Husbands, P., and Harvey, I. (1995) Noise and the Reality Gap: The Use of Simulation in Evolutionary Robotics. In F. Moran, A. Moreno, J.J. Merelo, P. Chacon (eds.) *Advances in Artificial Life: Proceedings of Third European Conference on Artificial Life.* Springer-Verlag, Heidelberg.

Miglino, O., and Lund, H. H. (1995) Robotics as an Educational Tool. Technical Report, Institute of Psychology, C.N.R., Rome.

Miglino, O., Nafasi, K., and Taylor, C. (1995) Selection for Wandering Behavior in a Small Robot. *Artificial Life* **2:1**, 101-116.

Mondada, F., Franzi, E., and Ienne, P. (1994) Mobile Robot miniaturisation: A tool for investigation in control algorithms. In *Lecture Notes Control and Information Science*, **200**, 501-514. Springer-Verlag, Heidelberg.

Pagliarini, L., Lund, H. H., Miglino, O., and Parisi, D. (1995) Artificial Life: A New Way to Build Educational and Therapeutic Games. Technical Report, Institute of Psychology, C.N.R., Rome.