Evolutionary Robotics: a Survey of Applications and Problems

Jean-Arcady Meyer, Phil Husbands and Inman Harvey,

Summary

This paper reviews evolutionary approaches to the automatic design of real robots exhibiting a given behavior in a given environment. Such a methodology has been successfully applied to various wheeled or legged robots, and to numerous behaviors including wall-following, obstacle-avoidance, light-seeking, or arena cleaning. Its potentialities and limitations are discussed in the text and directions for future work are outlined.

Introduction

In the last few years, several researchers have attempted to bypass the difficulties of hand-coding the control architectures of mobile robots that have to fulfil given missions in unknown, and possibly changing, environments. Because such difficulties stem from the impossibility of foreseeing each problem the robot will have to solve, and from the lack of basic principles upon which human design might rely, these researchers advocate the so-called *evolutionary robotics* approach, i.e., an automatic designing procedure. According to this approach, a robot's controller, and possibly its overall body plan, is progressively adapted to the specific environment and the specific problems it is confronted with, through an artificial selection process that eliminates ill-behaving individuals in a population while favoring the reproduction of better-adapted competitors.

Such a process calls upon some evolutionary procedure – like a *genetic algorithm* (Goldberg, 1989), an *evolution strategy* (Schwefel, 1995), or a *genetic programming* (Koza, 1992) approach. It involves a *genotype* - i.e., an information that evolves through successive generations - and a *phenotype* - i.e., the robot's control architecture, its body plan, and its behavior - that is encoded in the genotype. A dedicated *fitness function* is used to assess the behavior of each individual in the population and to direct the selection proper. Dedicated operators – such as *mutation* and *crossing-over* – give rise to new genotypes in the population and permit robots of ever-increasing fitness to be generated, until the process converges to some local or global optimum. In the majority of applications, the evolutionary procedure is performed in two stages: fitted phenotypes are first sought through simulation and are then downloaded in turn on a real robot to check their fitness with respect to real world constraints. However, in some other applications, the evolutionary procedure takes place within the robot's processor and fitnesses are directly assessed through real world interactions. In both cases, software controllers are evolved. They may be implemented as control programs – in high level language or in machine code -, as a variety of production-rule systems, or as neural networks. Finally, within the so-called *evolvable hardware* approach (Sanchez and Tomassini, 1996; Higuchi et al., 1997), genotypes code for the configuration of hardware controllers and body plans, and fitnesses are also assessed through real world interactions.

Although numerous aspects of the methodology of evolutionary robotics have been tested in simulation, such research efforts won't be cited in this review paper[1], which is centered on

[1] See, for instance, Husbands et al. (1994), Gomi and Griffith (1996), or Mataric and Cliff (1996).

real robot applications. The robot the most often used in the applications described herein is Khepera, but it will be shown that other robots, including walking robots, have been used as well. This paper will also provide a discussion of the current potentialities and limitations of evolutionary robotics and will end with suggestions for future work.

Real robot applications

A. Khepera

Khepera (Mondada et al., 1993) is a circular-shaped miniature mobile robot -- with a diameter of 55mm, a height of 30mm, and a weight of 70g – that is supported by two wheels and two small Teflon balls. In its basic configuration, it is equipped with eight infra-red proximity sensors – six on the front, two on the back – that may also act as visible-light detectors. The wheels are controlled by two DC motors with incremental encoder that move in both directions.

Using the Khepsim simulator, Jakobi et al. (1995) evolved both obstacle-avoidance and light-seeking behaviors in Khepera. The simulation was based on a continuous two-dimensional model of the real world physics and allowed to calculate the dynamics of the robot's sensory inputs in response to its motor signals. Recurrent networks of threshold units that were evolved in simulation evoked qualitatively similar behavior on the real robot, especially when the levels of noise present in the simulation had similar amplitudes to those observed in reality.

To evolve the capacity of moving in the environment while avoiding obstacles, Miglino et al. (Miglino et al., 1995a; Lund and Miglino, 1996) used a two-layer feedforward neural network with no hidden units and a fitness function with three components, which were respectively maximized by speed, by straight direction, and by obstacle avoidance. With the help of a genetic algorithm, the synaptic connections and thresholds of the neural controllers were first evolved through simulation. Then, the corresponding networks were downloaded on Khepera and proved to be efficient. A similar two-staged approach has been followed by Salomon (1996), who used a (3,6)-Evolution Strategy with self adaptation of the step size (Bäck and Schwefel, 1993). Likewise, Naito et al. (1997) used a genetic algorithm to configure how a set of 8 logic elements could be connected to each other and to the sensors and motors of the robot. Within this approach, each controller was downloaded on Khepera and its fitness was directly assessed in the real world. On the contrary, in Floreano and Mondada (1994), the whole evolutionary process took place entirely on the robot without human intervention, and two-layer Elman neural networks (Elman, 1990) were used as controllers. This architecture consisted of a single layer of synaptic weights from eight sensor units to two motor units, with recurrent connections within the output layer. Using the same neuronal architecture and the same fitness function, Floreano and Mondada (1996a) let evolve the type of the Hebbian rule that was employed by each synapse in the network. Each synapse was thus genetically described by a set of four properties: whether it was driving or modulatory, whether it was excitatory or inhibitory, its Hebbian rule, and its learning rate. Four Hebbian rules could be used: pure Hebbian, postsynaptic, presynaptic, and covariance (Willshaw and Dayan, 1990). Under such conditions, each decoded neural network changed its own synaptic strength configuration according to its genotypic specifications and without external supervision while Khepera interacted with its environment. Experimental results showed that the efficient

controllers that evolved exhibited synapses that were continuously changing in a dynamically stable regime. In other words, knowledge in such networks is not expressed by a final stable state of the synaptic configuration, but by a dynamical equilibrium. There are also indications that such plastic neurocontrollers are more resistant to sensor damage than standard static controllers.

Another study of the interactions between learning and evolution is that of Mayley who evolved simple feedforward neural controllers for wall-following in Khepera. In this work also, besides encoding the network's weights, the genome determined whether each weight was plastic or not – i.e., whether it might be changed or not by an Hebbian learning process. Experimental results indicated that, as long as there are costs to be paid for the ability to learn, learning is first selected for and then against as evolution progresses, thus illustrating how a learned trait or behavior may become genetically assimilated.

In Floreano and Mondada (1996b) the evolution of a set of behaviors that allowed a Khepera robot to locate a battery charger and periodically return to it so as to increase its chances of survival has been achieved. In this work, the Khepera robot was equipped with two additional sensors. One ambient light sensor was placed under the robot platform pointing downward, so as to detect a black painted area on the floor that was considered as place where its battery was recharged. Another simulated sensor was used to provide information about the current energy level of the robot's battery. Thus, the input layer of the neural network consisted of twelve receptors each clamped to one sensor: 8 for IR-emitted light, 2 for lateral ambient light, 1 for floor brightness, and 1 for battery charge. The controller architecture was completed with a hidden layer of 5 units with recurrent connections and an output layer of two units, one for each motor. To evaluate the fitness of each individual, each robot started its life with a fully charged battery that was discharged by a fixed amount at every time step and that was instantaneously recharged if the robot happened to pass over the black area. While a given maximum life time was allotted to each robot, a fully discharged battery entailed instantaneous death. The robot's fitness was accumulated at each step during evaluation and called upon two components: the first one was maximized by speed and the second by obstacle avoidance. Although such a fitness function did not specify neither the location of the battery station, nor the fact that the robot should reach it, the right behavior evolved because the accumulated fitness of each individual depended both on the performance of the robot and on the length of its life.

In the work of Nolfi (1996b) the parameters of a feedforward neural network with no hidden units were evolved to control a Khepera robot that had to explore his environment, to avoid walls and to remain close to a cylindrical target when it found it. The fitness of each controller was assessed through simulation and depended upon the time spent close to the target. Experimental results showed that the evolved individuals were successful in the real world and that, by intensively using an active perception strategy, they could overcome the problem posed by the fact that the walls and the target were hard to distinguish in most cases. As an extension of this work, and in order to study the interactions of individual learning and evolution, Nolfi and Parisi (1997) added two output units to such feedforward controllers. These units served as auto-teaching units (Nolfi and Parisi, 1993) that set the desired values of the two motor-controlling units when, at the beginning of each individual's test period, a backpropagation algorithm was activated. Because testing could be performed either in an environment with dark walls or in an environment with white walls, backpropagation made it possible for a given individual to learn in which environment it was placed and to accordingly

adjust during its lifetime the synaptic weights it inherited from the previous generation. Thus, through successive generations, individuals capable of learning more and more rapidly how to find the target did evolve.

Using simulations to evolve simple feedforward neurocontrollers that were later downloaded on a Khepera robot equipped with a gripper module, Nolfi (Nolfi and Parisi, 1995; Nolfi,1996a,1997a,b,c) evolved the task of keeping clear an arena surrounded by walls, in which small cylindrical trash objects were disposed at random. The best results were obtained when the neural controllers exhibited a so-called emergent modular architecture. Within such architecture, the number of available modules, their internal organization, and the mechanisms that determined their interaction were pre-designed and fixed. However, the way each of these modules was used at each time step depended upon the evolved values of each connection weight and bias within the overall architecture. Such values were directly binary encoded in individual genes. Fitnesses were evaluated by counting the number of objects correctly released outside the arena during a given evaluation time. During evolution, individuals capable of simply picking up targets were slightly favored. Likewise, experience showed that it was useful to artificially increase the number of times the robot encountered another target while carrying an object, in order to force the evolutionary process to select individuals able to avoid targets when the gripper was already holding something.

Researchers at Dortmund University (Nordin and Banzhaf, 1996; Banzhaf et al., 1997) evolved obstacle-avoidance and object-following behaviors in Khepera with a Genetic Programming (Koza, 1992) variant that manipulates machine code directly. Their system uses linear genomes composed of variable length strings of 32 bit instructions for a SUN-4 computer. Each instruction performs arithmetic or logic operations on a small set of registers and may also include a small integer constant of 13 bits at most. The genetic operators are tailored to manipulate genetic code directly. In particular, crossover occurs between instructions and thus changes the order and number of instructions in offspring programs; mutations are allowed to flip bits within instructions. To evolve obstacle avoidance, a fitness function with a negative and positive part was used. The former was the sum of all proximity sensors; the latter was dependent upon wheel speeds and assessed how straight and fast the robot was moving. For object following, the robot's task was to follow moving objects without colliding with them. The corresponding fitness function used values returned by the 4 sensors facing forward, and rewarded individuals capable of both moving towards objects far away and avoiding too close objects. Encouraging preliminary results have been obtained in experiments where the system is using a memory buffer that stores event vectors representing salient sensory-motor situations encountered in the past.

Instead of directly evolving a complex behavior as a whole, Lee et al. (1997a,b) evolved behavior primitives and behavior arbitrators for a Khepera robot that had to push a box toward a goal position indicated by a light source. To accomplish this task, they used a genetic programming system that evolved the controller programs of two behavior primitives, box pushing – keep pushing a box forward - and box-side-following – move along the side of a box. In addition, they also evolved an arbitrator program that was used to arrange the executing sequence of the behavior primitives. Experimental results show that controllers evolved in simulation were transferred to the real robot without loss of performance.

Several research efforts have aimed at evolving neural controllers for the Khepera robot through developmental approaches that call upon various biomimetic processes -- like cell division, cell differentiation, or cell adhesion -- to gradually build a neural control

architecture. Controllers for obstacle-avoidance, light-seeking or light-avoiding behaviors have thus been evolved by Eggenberger (1996). Wall-following and obstacle-avoidance behaviors have also been evolved through such a developmental approach by Michel (Michel, 1996; Michel and Collard, 1996).

Finally, with the aim of evolving a behavior that was at least one step up from the simple reactive behaviors that have been sought so far, Jakobi (1997a,b) succeeded to evolve reliably fit recurrent neural network controllers that allowed a Khepera robot to memorize on which side of a corridor it passed through a beam of light. Then, when the robot arrived at a T-maze junction at the end of the corridor, its task was to turn on the same side and move down the corresponding arm. Controllers that have been evolved within around 1000 generations in simulation were downloaded onto Khepera and performed the task satisfactorily and efficiently.

B. Other robots

Several experiments have been performed at Sussex University (Cliff et al., 1993; Harvey et al., 1994; Jakobi, 1997a,b) in which discrete-time dynamical recurrent neural networks and visual sampling morphologies are concurrently evolved to allow a gantry robot to perform various visually guided tasks. Such experiments called upon a CCD camera sensing its environment through a swiveling mirror. For instance, within an environment predominantly dark, the robot had to move toward fixed or mobile white targets. Likewise, it might had to approach a white triangle while ignoring a white rectangle. In such experiments, successful behaviors were evolved using a genetic algorithm acting on pairs of chromosomes encoding the visual morphology and the neural controller of the robot. One of the chromosomes was a fixed length bit string encoding the position and size of three visual receptive fields from which the visual signals processed by the neural controller were calculated. The other was a variable length character string encoding the number of hidden units and the number of excitatory and inhibitory connections between neurons. On the contrary, the number of input nodes was fixed to seven – one input for each of three visual receptive field and of for each of four tactile sensors – and the number of output nodes - whose signals were translated into gantry moves and mirror angular velocities - was fixed to four .

Work by Grefenstette and Schultz (1994) calls upon the use of the SAMUEL classifier system (Grefenstette and Cobb, 1991) for evolving collision-free navigation in a Nomad 200 mobile robot equipped with 20 tactile, 16 sonar, and 16 infra-red sensors. Within such an approach, besides from being possibly mutated, the condition part of each of SAMUEL's rule - which was compared against the current sensor readings – was also submitted to dedicated generalization and specialization operators. The task consisted of learning to reach a fixed goal location in a predetermined time, starting from a fixed initial position within an environment that contained obstacles whose positions were changed at each trial. With a population size of 50 rules, rule sets evaluated through simulation over 50 generations were downloaded on the robot and proved to be efficient 86% of the time. A similar approach is that of Colombetti and Dorigo (1993) who used the ALECSYS software tool (Dorigo, 1993) to evolve the control architecture of the AutonoMouse, a mouse-shaped autonomous robot equipped with two on/off eyes positioned in front of the robot and sensing light within a cone of about 60 degrees. In this work, the robot's control architecture was a set of interconnected classifier systems and the behavior to evolve was light-chasing. To succeed, the robot had to learn appropriate moves so as to cope with situations where the target light was on, but out of

the robot's sight. The robot's fitness was evaluated through light intensity, detected by a dedicated central light sensor.

Miglino et al. (1995b) evolved a four-layer Elman-like recurrent neural networks with 2 sensory units, 2 output units, 2 hidden units, and 1 memory unit that allowed a mobile Lego robot to explore the greatest percentage of an open area within an allotted number of steps. Two optosensors were used to detect whether the areas ahead and behind the robot's current location were black or white, thus allowing the robot to move within a central white surface surrounded by a black border. Such moves were determined by the values of the two output units. The architecture of the controllers was fixed and only the weights of the connections were encoded in the genotype, as a vector of 17 integer numbers. Although the fitness of each controller was assessed through simulations, experiments showed that evolved controllers were efficient in the real world, despite the fact that the real trajectories were significantly different from the simulated ones.

Yamauchi and Beer (1994) used a Nomad 200 mobile robot to let evolve neural controllers capable of identifying one of two landmarks based on the time-varying sonar signals received as the robot turned around the landmark. The robot's trajectory was controlled by a fixed behavior-based control system that allowed the robot to find a wall and follow it counterclockwise around the perimeter of the experimental room. A single sonar on the left side of the robot was used to detect a central landmark and its range signals were input to each of the eight neurons in a continuous-time fully-connected recurrent neural network. One of these neurons was designated the output unit and its firing rate after a fixed period of time – i.e., after the input signal sequence has been integrated over time - was used to classify the landmark. Network parameters - like time constants, thresholds, or connection weights - were genetically encoded as vectors of real numbers, of which each element was indivisible under crossover. The fitness function of each individual in a population of 100 networks was evaluated in simulation and assessed the average capacity of the network to correctly identify the landmarks over six test trials. After 15 generations, an individual capable of correctly recognizing the landmarks in simulation was generated. When transferred on the real robot, it correctly classified the landmarks in 17 out of 20 test trials.

In Yamauchi (1993), other evolutionary robotic simulations are described that have been successfully applied to predator avoidance in a Nomad 200 robot. In this approach, dynamic neural networks were used to perform the task of evading a moving pursuer while avoiding collisions with stationary obstacles.

Baluja (1996) presents an evolutionary method for designing a neural controller for the Carnegie Mellon's NAVLAB autonomous land vehicle. To assess its steering abilities, the neural network is shown video images from the NAVLAB's onboard camera as a person drives and its task is to output the direction in which the person is currently steering. A maximal network architecture is defined, which determines the structure and maximum connectivity of the controller to which, during evolution, connections may be removed but not added. In one series of experiments, this maximal network architecture was a fully-connected perceptron with a 15 x 16 pixels input retina, a five unit hidden layer, and a single unit layer whose activation determined how sharply the steering should be to the left or to the right of center. In a second series of experiments, the same architecture was used, but with 30 output units, each of which being considered as representing the network's vote for a particular

steering direction. In both cases, the so-called PBIL (Population-Based Incremental Learning) evolutionary algorithm was used, according to which a probability vector is evolved as a prototype from which potentially highly fit networks can be derived. This vector specifies the probabilities of having a 1 or a 0 in each bit position of a string encoding the topology and connection weights of a neural controller. During evolution, in a manner similar to the training of a competitive learning network, the values in the probability vector are progressively shifted toward the bit values that specify efficient network designs. This evolutionary approach performed better, on average, than standard backpropagation, especially in the one-output networks.

Using a genetic algorithm acting on individuals represented as real-coded vectors of weights, Meeden (1996) evolved recurrent neural controllers for a four-wheeled robot that had to continually keep moving, to avoid contacts with walls, and either to seek or avoid light depending upon its current goal. This robot was equipped with three front and one back touch sensors, with two light sensors, and with one goal sensor that indicated that the robot should seek out (respectively avoid) the light until a maximum (respectively minimum) light reading was obtained. For movement, the robot had two servo-motors: one controlling forward and backward motion, the other controlling steering. Elman-like networks with a fixed architecture were used for that purpose - with 7 input units each connected to a given sensor, 5 hidden units with recurrent self-connections, and 4 output units that determined how to set the motors for the next time step. During evaluation, the fitness of a given controller was incremented or decremented after each robot's action, according to a reward scale that took into account whether or not the robot accomplished a light goal, kept moving, had any touch sensor triggered, and correctly followed the light gradient. Experimental results showed that the evolutionary update of weights out-performed a complementary reinforcement backpropagation learning algorithm (Ackley and Litman, 1990) under delayed reinforcement conditions, i.e., when no light gradient reinforcement was provided between two switching-goal episodes.

Jeong and Lee (1997) got promising results suggesting that a genetic algorithm could be used to automatically design the controllers and the control strategies for two-wheeled soccer playing robots. Such robots are assumed to be used within an experimental setup consisting of a host computer that processes the vision data acquired by a camera and sends to each robot information about the playground positions of the ball and of each robot. A two-stage evolutionary approach has been investigated. In a first stage, production rules have been evolved, whose condition parts take into account the positions of the relevant objects – i.e., the partners, the opponents, the goals, and the ball – and whose action parts trigger a relevant action – i.e., a move, a dribble or a kick. In a second stage, optimal on-off control signals to the motors were evolved that allowed a robot to reach a position with desired coordinates and orientation.

Gallagher et al. (1996) describe experiments were neural networks controlling locomotion in an artificial insect were evolved in simulation and then successfully downloaded on a real 6-legged robot. In this approach, each leg was controlled by a fully interconnected network of 5 Hopfield-like continuous neurons (Hopfield, 1984), each receiving a weighted sensory input from that leg's angle sensor. Three of these neurons were motor neurons that respectively governed the state of the forward and backward joint torques of the leg and the state of the corresponding foot. The remaining two neurons were interneurons with no pre-specified role. Thanks to various simplifying assumptions (Beer and Gallagher, 1992), a set of only 50 parameters – which described neuronal physical constants, crossbody connection weights and

intersegmental connection weights – needed to be encoded in the insect's genotype as mere bit strings.

A genetic algorithm has been used by Galt et al. (1997) to derive the optimal gait parameters for a Robug III robot – an 8-legged, pneumatically powered walking and climbing robot. The individual genotypes were encoded to represent the phase and duty factors – i.e., the coordinating parameters that represent each leg's support period and the time relationships between the legs. Controllers were thus evolved that have been proved capable of deriving walking gaits that are suitably adapted to a wide range of terrains, damages or system failures. Future research will be targeted at using information on the terrain contours provided by the robot's legs. Such information can be used by neural networks to provide one step ahead forecast of the terrain conditions and hence improve the walking efficiency.

Gomi and Ide (1997a,b) evolved the gaits of a 8-legged OCT-1 robot (AAI Systems, Inc.) by loading it with a set of 50 software invoked control processes that are each given in turn a fixed amount of time to actuate the robot's legs. The corresponding genotypes are made of 8 similarly organized sets of genes, each gene coding for legs motion characteristics like the amount of delay after which the leg begins to move, the direction of the leg's motion, the end positions of both vertical and horizontal swings of the leg, the vertical and horizontal angular speed of the leg, etc. The fitness function is set in favor of a robot that stands up, evolves coordination among its legs motions, and has a tendency to move forward. Moreover, fitness scores are increased when internal sensors monitoring the servo motor electric currents indicate that a given leg is moved under proper loading conditions. Fitness scores are decreased when any of the sensors located on the belly of the robot detects a contact with the floor. Typically, after generation 10, most individuals succeed in standing and walking with a faint gait. Likewise, after a few dozen generations, a mixture of tetrapod and wave gaits is obtained.

Gruau and Quatramaran (1997) also evolved controllers for walking in the OCT-1 robot. Using a developmental approach called Cellular Encoding (Gruau, 1995) – i.e., an approach that genetically encodes a grammar-tree program that controls the division of cells growing into a discrete-time dynamical recurrent neural network – they first evolved a single-leg neural controller with one input and two outputs. When commands for return stroke or power stroke were input to the controller, it succeeded to respectively lift the foot up and propel the leg forward or to left the foot down and propel the leg backward. Then, they put together 8 copies of the leg controller and evolved a neural network that called upon 8 oscillators with correct frequency, coupling, and synchronization, which generated a smooth and fast quadripod locomotion gait.

C. Evolvable Hardware

Evolved Hardware controllers are not *programmed* to follow a sequence of instructions, they are *configured* and then allowed to behave in real time according to semiconductor physics.

Thompson (1995,1997) used artificial evolution to design a FPGA (Field Programmable Gate Array) hardware circuit as an on-board controller for a two-wheeled autonomous mobile robot displaying simple wall-avoidance behavior in an empty arena. A FPGA is a Very Large Scale Integration (VLSI) silicon chip containing a large array of components and wires. Switches

distributed throughout the chip can be set by an evolutionary algorithm and determine how each component behaves and how it connects to the wires. Thompson's approach called upon a so-called DSM (Dynamic State Machine) equipped with genetic synchronizers and with a global clock whose frequency was also under genetic control. Thus evolution determined whether each signal was passed straight through asynchronously, or whether it was synchronized according to the global clock. This process took place within the robot in a kind of "virtual reality" in the sense that the real evolving hardware controlled the real motors, but the wheels were just spinning in the air. The movement that the robot would have actually performed if the wheels had been actually supporting it were then simulated and the sonar echo signals that the robot was expected to receive were supplied in real time to the hardware DSM. Excellent performances were attained after 35 generations, with good transfer from the virtual environment to the real world. Similar results have been obtained with a Khepera robot equipped with an onboard Field-Programmable Gate Array (FPGA) (Thompson, 1997).

Using a Boolean function approach implemented on gate-level evolvable hardware, Keymeulen et al. (1997a,b) evolved a navigation system for a mobile robot capable of reaching a colored ball while avoiding obstacles during its motion. The mobile robot was equipped with infra-red sensors and an active vision system furnishing the direction and the distance to the colored target. A programmable logic device (PLD) was used to implement a Boolean function in its disjunctive form, which has been proved to be sufficient to control tracking-avoiding tasks (Lund and Hallam, 1997). It appeared that such gate level evolvable hardware was able to take advantage of the correlations in the input states and to exhibit useful generalization abilities, thus allowing the simulated evolution of a robust behavior in simple environments and a good transfer into the real world. Future work aims at accelerating on-line evolution by allowing the robot to do some experimentation in an internal model of its environment, to be implemented in an additional special purpose evolvable system.

Finally, Lund et al. (1997) advocate the use of so-called *true evolvable hardware* to evolve, not only a robot's control circuit, but also its body plan, which might include the types, numbers and positions of the sensors, the body size, the wheel radius, the motor time constants, etc. These authors are currently developing a new piece of reconfigurable hardware that will make it possible to co-evolve the control mechanisms and the auditory morphology of a Khepera robot behaving like a female cricket which is able to use phonotaxis to locate a song emitting male.

Discussion

Conclusion

References

Ackley, D.H. and Littman, M.L. 1990. Generalization and scaling in reinforcement learning. In Touretzky (Ed.). *Advances in Neural Information Processing Systems 2*. Morgan Kaufmann.

Bäck, T. and Schwefel, H.P. 1993. An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation*, 1,1, 1-23.

Baluja, S. 1996. Evolution of an Artificial Neural Network Based Autonomous Land Vehicle Controller. *IEEE Transactions on Systems, Man, and Cybernetics* – Part B: Cybernetics. 26, 3, 450-463.

Banzhaf, W., Nordin, P. and Olmer, M. 1997. Generating Adaptive Behavior for a Real Robot using Function Regression within Genetic Programming. In Koza et al. (Eds.). *Genetic Programming 1997*. Morgan Kaufmann.

Beer, R.D. and Gallagher, J.C. 1992. Evolving Dynamical Neural Networks for Adaptive Behavior; *Adaptive Behavior*, 1, 1, 91-122.

Cliff, D., Harvey, I. and Husbands, P. 1993. Explorations in evolutionary Robotics. *Adaptive Behavior*. 2,1, 73-110.

Colombetti, M. and Dorigo, M. 1993. Learning to Control An Autonomous Robot By Distributed Genetic Algorithms. In Meyer, Roitblat and Wilson (Eds.). *Proceedings of the Second International Conference on Simulation of Adaptive behavior: From Animals to Animats 2*. The MIT Press/Bradford Book.

Dorigo, M. 1993. Genetic and non-genetic operators in ALECSYS. *Evolutionary Computation*. 1(2), 151-164.

Eggenberger, P. 1996. Cell Interactions as a Control Tool of Developmental Processes for Evolutionary Robotics. In Maes, Mataric, Meyer, Pollack and Wilson (Eds.). *Proceedings of the Fourth International Conference on Simulation of Adaptive behavior: From Animals to Animats 4.* The MIT Press/Bradford Book.

Elman, J.L. 1990. Finding structure in time. *Cognitive Science.* 2, 179-211.

Floreano, D. and Mondada, F. 1994. Automatic Creation of an Autonomous Agent: Genetic Evolution of a Neural-Network Driven Robot. In Cliff, Husbands, Meyer and Wilson (Eds.). *Proceedings of the Third International Conference on Simulation of Adaptive behavior: From Animals to Animats 3.* The MIT Press/Bradford Book.

Floreano, D. and Mondada, F. 1996a. Evolution of plastic neurocontrollers for situated agents. In Maes, Mataric, Meyer, Pollack and Wilson (Eds.). *Proceedings of the Fourth International Conference on Simulation of Adaptive behavior: From Animals to Animats 4.* The MIT Press/Bradford Book.

Floreano, D. and Mondada, F. 1996b. Evolution of Homing Navigation in a Real Mobile Robot. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics.* 26, 3, 396-407.

Gallagher, J.C., Beer, R.D., Espenschield, K.S. and Quinn, R.D. 1996. Application of evolved locomotion controllers to a hexapod robot. *Robotics and Autonomous Systems.* 19, 95-103.

Galt, S., Luk, B.L. and Collie, A.A. 1997. Evolution of Smooth and Efficient Walking Motions for an 8-Legged Robot. *Proceedings of the 6th European Workshop on Learning Robots.* Brighton, UK.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley.

Gomi, T. and Griffith, A. 1996. Evolutionary Robotics – An Overview. *Proceedings of the IEEE 3rd International Conference on Evolutionary Computation.* IEEE Society Press.

Gomi, T. and Ide, K. 1997a. Emergence of gaits of a legged Robot by Collaboration through Evolution. *Proceedings of the International Symposium on Artificial Life and Robotics.* Springer Verlag.

Gomi, T. and Griffith, A. 1996. Evolutionary Robotics – An Overview. *Proceedings of the IEEE 3rd International Conference on Evolutionary Computation.* IEEE Society Press.

Grefenstette, J. and Cobb, H.C. 1991. *User's guide for SAMUEL* (NRL Memorandum Report 6820). Washington, DC: Naval research Laboratory.

Grefenstette, J. and Schultz, A. 1994. An evolutionary approach to learning in robots. *Proceedings of the Machine Learning Workshop on Robot Learning.* New Brunswick, NJ.

Gruau, F. 1995. Automatic definition of modular neural networks. *Adaptive Behavior*. 3, 2, 151-183.

Gruau, F. and Quatramaran, K. 1997. Cellular Encoding for Interactive Evolutionary Robotics. In Husbands and Harvey (Eds.). *Fourth European Conference on Artificial Life.* The MIT Press/Bradford Books.

Harvey, I., Husbands, P. and Cliff, D. 1994. Seeing The Light: Artificial Evolution, Real Vision. In Cliff, Husbands, Meyer and Wilson (Eds.). *Proceedings of the Third International Conference on Simulation of Adaptive behavior: From Animals to Animats 3.* The MIT Press/Bradford Book.

Harvey, I., Husbands, P., Cliff, D., Thompson, A. and Jakobi, N. 1997. Evolutionary Robotics: The Sussex Approach. *Robotics and Autonomous Systems.* 20, 205-224.

Higuchi, T., Iwata, M. and Liu, W. (Eds.). 1997. *Evolvable Systems: From Biology to Hardware.* Springer Verlag.

Hopfield, J.J. 1984. Neurons with graded response properties have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences, USA,* 81, 3088-3092.

Husbands, P, Harvey, I., Cliff, D. and Miller, G. 1994. The Use of Genetic Algorithms for the Development of Sensorimotor Control Systems. In Nicoud and Gaussier (Eds.). *From Perception to Action.* IEEE Computer Society Press.

Jakobi, N. 1997a. Half-baked, Ad-hoc and Noisy: minimal Simulations for Evolutionary Robotics. In Husbands and Harvey (Eds.). *Fourth European Conference on Artificial Life.* The MIT Press/Bradford Books.

Jakobi, N. 1997b. Evolutionary Robotics and the Radical Envelope of Noise Hypothesis. *Adaptive Behavior*. 6,1, 131-174.

Jakobi, N., Husbands, P. and Harvey, I. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In Moran, Moreno, Merelo and Chacon (Eds.). *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life.* Springer Verlag.

Jeong, I.K. and Lee, J.J. 1997. Evolving cooperative mobile robots using a modified genetic algorithm. *Robotics and Autonomous Systems,* 21, 197-205.

Keymeulen, D., Durantez, M., Konaka, M., Kuniyoshi, Y. and Higuchi, T. 1997a. An Evolutionary Robot Navigation System Using a Gate-Level Evolvable Hardware. In Higuchi, Iwata and Liu (Eds.). *Evolvable Systems: From Biology to Hardware.* Springer.

Keymeulen, D., Konaka, K., Iwata, M., Kuniyoshi, Y. and Higuchi, T. 1997b. Robot Learning using gate-Level evolvable hardware. *Proceedings of the 6th European Workshop on Learning Robots.* Brighton, UK.

Koza, J. 1992. Genetic Programming. The MIT Press.

Lee, W.P., Hallam, J. and Lund, H.H. 1997a. Applying Genetic Programming to Evolve Behavior Primitives and Arbitrators for Mobile Robots. *Proceedings of IEEE Fourth International Conference on Evolutionary Computation.* Piscataway, NJ.

Lee, W.P., Hallam, J. and Lund, H.H. 1997b. Learning Complex Robot Behaviours by Evolutionary Approaches. *Proceedings of the 6th European Workshop on Learning Robots.* Brighton, UK.

Lund, H.H. and Hallam, J. 1997. Evolving sufficient robot controllers. *Proceedings of IEEE Fourth International Conference on Evolutionary Computation*. Piscataway, NJ.

Lund, H.H., Hallam, J. and Lee, W.P. 1997. Evolving Robot Morphology. *Proceedings of IEEE Fourth International Conference on Evolutionary Computation.* Piscataway, NJ.

Lund, H.H. and Miglino, O. 1996. From Simulated to Real Robots. *Proceedings of the 3rd IEEE International Conference on Evolutionary Computation.* IEEE Computer Society Press.

Mataric, M. and Cliff, D. 1996. Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems.* 19, 67-83.

Mayley, G. 1996. The Evolutionary Cost of Learning. In Maes, Mataric, Meyer, Pollack and Wilson (Eds.). *Proceedings of the Fourth International Conference on Simulation of Adaptive behavior: From Animals to Animats 4.* The MIT Press/Bradford Book.

Meeden, L.A. 1996. An Incremental Approach to Developing Intelligent Neural Network Controllers for Robots. *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics.* 26, 3, 474-485.

Michel, O. 1996. An Artificial life Approach for the synthesis of Autonomous Agents. In Alliot, Lutton, Ronald, Schoenauer and Snyers (Eds.). *Artificial Evolution*. Springer.

Michel, O. and Collard, P. 1996. Artificial Neurogenesis: An application to Autonomous Robotics. In Radle (Ed.). *Proceedings of The 8th. International Conference on Tools in Artificial Intelligence.* IEEE Computer Society Press.

Miglino, O., Lund, H.H. and Nolfi, S. 1995a. Evolving Mobile Robots in Simulated and Real Environments. *Artificial Life*, 2, 417-434.

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

Mondada, F., Franzi, E. and Ienne, P. 1993. Mobile robot miniaturization: A tool for investigation in control algorithms. *Proceedings of the Third International Symposium on Experimental Robotics.* Kyoto, Japan.

Naito, T., Odagiri, R., Matsunaga, Y., Tanifuji, M. and Murase, K. 1997. Genetic Evolution of a Logic Circuit Which Controls an Autonomous Mobile Robot. In Higuchi, Iwata and Liu (Eds.). *Evolvable Systems: From Biology to Hardware.* Springer.

Nolfi, S. 1996a. *Evolving non-Trivial Behaviors on Real Robots: a garbage collecting robot.* Technical Report, Institute of Psychology, CNR, Rome.

Nolfi, S. 1996b. *Adaptation as a more powerful tool than decomposition and integration.* Technical Report, Institute of Psychology, CNR, Rome.

Nolfi, S. 1997a. Using Emergent Modularity to Develop Control Systems for Mobile Robots. *Adaptive Behavior*. 5,3/4, 343-363.

Nolfi, S. 1997b. Evolving Non-Trivial Behavior on Autonomous Robots: Adaptation is More Powerful Than decomposition and Integration. In Gomi (Ed.). *Evolutionary Robotics. From Intelligent Robots to Artificial Life (ER'97).* AAI Books.

Nolfi, S. 1997c. Evolving non-Trivial Behaviors on Real Robots: a garbage collecting robot. *Robotics and Autonomous Systems*. In Press.

Nolfi, S., Floreano, D., Miglino, O. and Mondada, F. 1994. How to evolve autonomous robots: Different approaches in evolutionary robotics. In Brooks and Maes (Eds.). *Artificial Life IV*. The MIT Press/Bradford Books.

Nolfi, S. and Parisi, D. 1993. Auto-teaching: Networks that develop their own teaching input. In Deneubourg, Bersini, Goss, Nicolis and Dagonnier (Eds.). *Proceedings of the Second European Conference on Artificial Life.* Free University of Brussels.

Nolfi, S. and Parisi, D. 1995. Evolving non-trivial behaviors on real robots: an autonomous robot that picks up objects. In Gori and Soda (Eds.). *Topics in Artificial Intelligence. Proceedings of the Fourth Congress of the Italian Association for Artificial Intelligence.* Springer.

Nolfi, S. and Parisi, D. 1997. Learning to Adapt to Changing Environments in Evolving Neural Networks. *Adaptive Behavior*, 5, 1, 75-98.

Nordin, P. and Banzhaf, W. 1996. An On-Line Method to Evolve Behavior and to Control a Miniature Robot in Real Time with Genetic Programming. *Adaptive Behavior*, 5, 2, 107-140.

Salomon, R. 1996. Increasing Adaptivity through Evolution Strategies. In Maes, Mataric, Meyer, Pollack and Wilson (Eds.). *Proceedings of the Fourth International Conference on Simulation of Adaptive behavior: From Animals to Animats 4.* The MIT Press/Bradford Book.

Sanchez, E. and Tomassini, M. (Eds.). 1996. *Towards Evolvable Hardware. The Evolutionary Engineering Approach.* Springer Verlag.

Schwefel, H.P. 1995. *Evolution and Optimum Seeking*. Wiley.

Thompson, A. 1995. Evolving electronic robot controllers that exploit hardware resources. In Moran, Moreno, Merelo and Chacon (Eds.). *Advances in Artificial Life: Proceedings of the Third European Conference on Artificial Life.* Springer Verlag.

Thompson, A. 1997. Artificial Evolution in the Physical World. In Gomi (Ed.). *Evolutionary Robotics. From Intelligent Robots to Artificial Life (ER'97).* AAI Books.

Willshaw, D and Dayan, P. 1990. Optimal plasticity from matrix memories: What goes up must come down. *Neural Computation*, 2, 85-93.

Yamauchi, B. 1993. *Dynamical neural networks for mobile robot control*. Naval Research Laboratory Memorandum Report AIC-033-93. Washington.

Yamauchi, B. and Beer, R. 1994. Integrating Reactive, Sequential, and Learning Behavior Using Dynamical Neural Networks. In Cliff, Husbands, Meyer and Wilson (Eds.). *Proceedings of the Third International Conference on Simulation of Adaptive behavior: From Animals to Animats 3.* The MIT Press/Bradford Book.