Multilevel Darwinist Brain and Autonomously Learning to Walk.

F. Bellas[1], A. Lamas[2], R. J. Duro[3]
Grupo de Sistemas Autónomos
Universidade da Coruña
Mendizábal s/n, 15403 Ferrol
SPAIN
[1]fran@cdf.udc.es [2]alamas@cdf.udc.es [3]richard@udc.es

Abstract

This paper is devoted to the application of the Multilevel Darwinist Brain to the autonomous learning and generation of gait patterns by a hexapod robot. This robot must learn to walk and reach an objective without any pre-programmed fitness function or behaviour. It develops a model of its world and of itself by interacting with the world, very much like a child, and this allows it to choose the appropriate actions to fulfil its motivations. As this interaction progresses its actions in the world become more focused and efficient and after some time it is able to generate the optimal gait and control the swing amplitude of the legs so as to reach its objective every time, independently of where it is located with respect to its body. To achieve these behaviours we have developed a Darwinist cognition mechanism operating over artificial neural networks that allows the robot to learn from its interaction with the world generating original solutions and making use of experience.

1. Introduction

The Multilevel Darwinist Brain (MDB) is a proposal for a Darwinist cognitive mechanism intended for autonomous artificial organisms that must generate original solutions and use previously acquired experience when negotiating new environments and situations. The initial ideas for MDB were introduced in [1], [2] and [3] and in this paper we will present some results of its implementation on a real robot that learns to walk and reach an objective very much like a child learns to perform actions, by extracting models and structure from its interactions with the world.

This cognitive mechanism is based on a series of theories within the field of cognitive science that relate the brain and its operation through a Darwinist process. These theories are: the Theory of Evolutionary Learning Circuits (TELC) [4], [5]; the Theory of Selective Stabilization of Synapses (TSSS) [6], [7]; the Theory of Selective Stabilization of Pre-Representations (TSSP) [8] and the Theory of Neuronal Group Selection (TNGS) or "Neural Darwinism" [9]. Each theory has its own features but they all lead to the same concept of cognitive structure based on the fact that the brain adapts its neural connections in real time through evolutionary or selectionist processes.

Taking inspiration from these theories we have developed an operational cognitive mechanism for artificial organisms which does not predetermine their behavior or learning and which allows them to find creative and original solutions in an autonomous way.

Some other authors have made reference to structures that make use of on-line evolutionary techniques, for example [10] and [11], applied to autonomous robotics.

The paper is structured as follows: in sections 2 and 3 we will review the main features of the mechanism and its operation. Section 4 is divided in 3 parts: in section 4.1 we present the real robot and the model used in simulation, in section 4.2 we provide the first example of application, where the hexapod robot must learn to walk, and in section 4.3 the second example where it must reach an objective. Finally, in section 5 we indicate some conclusions and the work we will be carrying out in the near future.

2. Basis for autonomous cognition

When constructing a cognitive mechanism for an autonomous agent we must establish the conditions for the interaction of the agent with the world in a very clear way. Firstly, one must consider that to carry out any task, there must be a motivation that guides the behaviour. This motivation must somehow reflect how the organism feels when it acts on its environment. From this basic idea, we must now provide a series of definitions:

- The *external perception* s(t) of an agent is made up of the sensorial information it is capable of acquiring from the environment in which it operates through its sensors.
- The *internal perception* i(t) of an agent consists of the sensorial information it has on its internal feelings.
- We define the *global perception* P(t) of the agent, which is made up of the external perception s(t) and the internal perception i(t).
- The *internal state* I(t) establishes the degree to which the motivation or motivations are satisfied.

Thus, if we consider the dependencies of the different elements that come into play, we have that the internal state must be a function $\mathcal{I}$ of the global perception, and, consequently, of the internal and external perceptions:

$$I(t) = \mathcal{I}[P(t)] = \mathcal{I}[s(t), i(t)]$$

On the other hand, the external perception depends on the last action performed by the agent A(t) and on the sensorial perception it had of the external world in the previous time instant s(t-1):

$$s(t) = \mathcal{W}[s(t-1), A(t)]$$

Consequently, we have that the internal state of the agent may be expressed as a function of the sensorial inputs in time instant t-1, of the action applied in the current instant and of the internal perception in the current instant:

$$I(t) = \mathcal{I}[s(t), i(t)] = \mathcal{I}\{\mathcal{W}[s(t-1), A(t)], i(t)]\}$$

The interaction of the agent with its environment must lead to the satisfaction of the motivation, which, without any loss of generality, may be expressed as the maximization of the internal state. Thus:

$$\max [I(t)] = \max \left(\mathcal{I}\{\mathcal{W}[s(t-1), A(t)], i(t)]\}\right)$$

The only parameter that may be modified during the maximization process is the action, as the external perception cannot be manipulated (unless we change the environment in order to facilitate the behaviour). That is, the cognitive mechanism must explore the possible action space in order to maximize the internal state. This implies the use of a search algorithm such as a traditional gradient descent algorithm, a genetic algorithm, etc. For this search process to be meaningful, we must obtain functions $\mathcal{I}$ and $\mathcal{W}$ from the data made available by interacting with the world, or no maximization process is possible. To find these functions we again use a search algorithm but, in this case, in a functional space. Thus, we require three search algorithms, one that seeks the best action and two that permit choosing the best functions $\mathcal{I}$ and $\mathcal{W}$ .from the data obtained by interacting with the environment.

We will call these functions *Internal Model* ($\mathcal{I}$) and *World Model* ($\mathcal{W}$). These functions may be implemented using different underlying mechanisms. In our work we have opted for the use of Artificial Neural Networks due to the fact that they are universal function approximators.

3. Multilevel Darwinist Brain

To implement the concepts explained in the previous section in a computationally efficient way, we have considered *Strategies* as sequences of actions applied to the effectors of the agent. An ideal strategy would depend on the current external perceptions, internal state and motivations. On the other hand, as stated before, a *World model* ($\mathcal{W}$) is a function that relates the sensory inputs of the agent in the instant of time t with the sensory inputs in the instant t+1 after applying a strategy. A world model permits evaluating the consequences of actions on the environment as perceived by the agent (sensed values in t+1). An *Internal model* ($\mathcal{I}$) is a function that relates the sensory inputs in instant of time t+1 with the internal state according to the motivation for the agent. An internal model permits predicting the internal state in instant t+1 from the sensory inputs for t+1 and the motivation. That is, it provides an indication of a satisfaction factor the agent uses to make decisions. Finally, an a*ction-perception pair* is a set of values made up by the sensorial inputs and the internal state obtained after the execution of a strategy in the real world. It is used when perfecting world and internal models.

The problem is how to connect these elements into an on line mechanism that permits the agent to generate original solutions making use of previous experience, and without the intervention of a designer, so that the motivations are fulfilled.

The way we have solved the problem is shown in the block diagram of Fig. 1, which represents the whole mechanism. The final objective of the mechanism is to obtain the strategy the agent must execute in the real world to satisfy its motivations. This selected strategy is represented in the block diagram of Fig. 1 by the block labeled *Current Strategy* and is applied to the *Environment* through the *Effectors* providing new *Sensing* values for the agent. Thus, after each iteration, we have an action-perception pair obtained from the real world. These action-perception pairs are stored in the *Short-Term Memory* and are used as the fitness function for the evolution of the world and internal models.

We have three concurrent evolutionary processes in our mechanism that are represented in Fig. 1:

- *World Model Evolver:* structure that manages world models through evolution. This structure evolves a population of world models and each moment of time selects the best one according to the information it has available about the real world.
- *Internal Model Evolver:* manages an internal model base, evolving it and providing the best internal model available when it is requested.
- *Strategy Evolver:* manages and evolves strategies and when the agent needs one it provides it.

Each one of these evolutionary processes starts from the population stored in a memory (*World Model Memory, Internal Model Memory and Strategy Memory*). The contents of these memories are random for the first iteration of the mechanism. These contents are maintained from one iteration to the next in order to obtain a generally good population of models and not a superindividual. The individual with the highest fitness value after evolution according to the information of the short-term memory is selected as the current model, both in the case of world and internal models. These current models are represented in Fig. 1 by the *Current World Model* and *Current Internal Model* blocks .

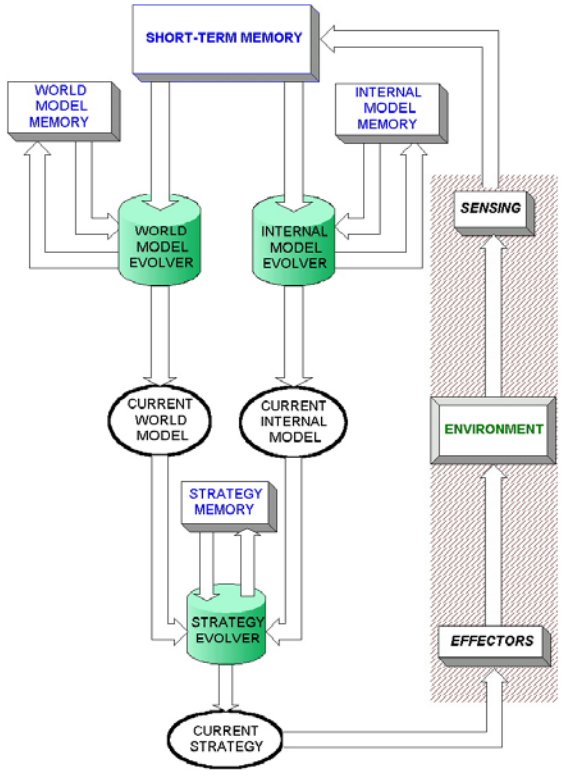

Fig. 1. Block Diagram of the cognitive mechanism where the shadowed area represents the external operation of the mechanism and the remaining area represents the internal operation.

In the case of strategies, these are tested during their evolution by implementing a virtual environment using the current models, that is, the current world model provides de sensory inputs in instant t+1 for the strategy we want to test. The current internal model uses these predicted values as input to predict the internal state corresponding to the strategy according to the agent's motivation. The strategy with the highest fitness value is selected as the *Current Strategy* and is applied to the *Environment* through the *Effectors*, obtaining again new *Sensing* values and thus a new *action-perception pair* that is stored in the *Short-Term Memory*.

This basic cycle is repeated and, as time progresses, the models become better adapted to the real world and the selected strategies improve in their efficiency to fulfil the agent's motivations.

As we display in Fig. 1, the operation of the mechanism can be divided into two parts: first the *internal operation*, which may be assimilated to a thought process. That is, during this stage, and without any interaction with the external world, the mechanism explores possible solutions to its problems. It uses the information it has compiled through its models and what it knows of the environment to obtain better models, deciding in the end the best strategy to employ. In *external operation* the selected strategies are applied in the environment, producing new values the sensors can measure and thus, producing new information the agent can use to improve its models of the world and of itself.

Consequently, we can say that the environment through the sensory inputs evaluates the world models and the internal models. The current model through the predicted satisfaction of the motivations evaluates the strategies.

4. Application to real robots

We have implemented the mechanism presented above in a hexapod robot that must learn to walk and once it does, it must learn to reach an object applying the acquired gait.

4.1. The robot and the simulator

In the left image of figure 2 we show the hexapod robot, Hermes II, used in the examples. It is a robust robot provided with six infrared sensors placed in the top of each leg, two whiskers, inclinometers and six force sensors.

The mechanism was applied to a simulated model of the Hermes II robot (created using the DADS a 3-D mechanical simulator that is shown in the right image of figure 2) and then transferred to the real robot.

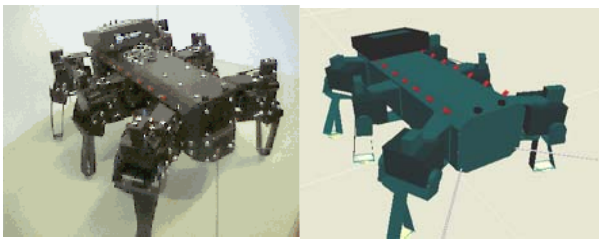

Fig. 2. The left image shows the Hermes II robot and the right one shows the model used in simulation.

4.2. Learning to walk

In this example, we want the Hermes robot to learn to walk. We can describe the motion of each leg through 3 parameters (for the swing and lift motion): the initial phase, that establishes the starting point of the leg motion, the frequency, that increases or decreases the speed of the movement and the amplitude of sweep, that permits the robot to turn. In this case, all the parameters are fixed except the initial phase of the swing motion. The different combinations of phases lead to different ways of walking (gaits), some useful, some useless and some even completely impractical, and we hope that the mechanism will allow the robot to develop an efficient gait so that it can fulfil its motivations.

The robot starts from point (0,0) in each iteration and an object (a cone) is placed in point (0,1), where distances are measured in meters. The mechanism selects the gait that must be applied and the robot uses it during a fixed time (13 seconds in simulation, 24 seconds in the real robot). Using the infrared sensors, the robot knows the distance to the cone all the time and these values are used as the input sensed values to the world model.

The infrared sensors provide poor information in real time, so we have applied the virtual sensor presented in [12] to each leg. This way, we have a world model with 7 inputs, the distance to the cone provided by the virtual sensor and the 6 input phases applied to the legs. The output from the world model is the predicted distance to the cone. We have implemented the world models using artificial neural networks as explained before, and depending on the size of these networks the evolution was different. The best results were obtained using two hidden layers with 4 neurons each. In figure 4 we show the evolution of the mean squared error between the distance predicted by the world model and the real one.
As we can see, the error becomes very small about

Fig. 4. Evolution of the mean squared error between the output of the world model (distance to the object) and the real one.

iteration 180 but it oscillates. This is because a world model that is evolved for a given set of contents in the short-term memory, could be bad for another. In this example, the short-term memory contains 40 action-perception pairs and, as we explained before, it works like a FIFO memory.
The internal state is, in this case, directly the predicted distance to the cone, so the Hernes robot must select the gait in order to minimize this distance.
For the evolution of the world models we have used a genetic algorithm with 700 individuals, 57 genes (corresponding to the weights and bias of the 7-4-4-1 neural network), 60% of crossover and 2% of mutation.
For the evolution of the strategies we have used a genetic algorithm with 120 individuals, 6 genes (direct encoding of the input phases), 60% of crossover and 6% of mutation.
In order to test how good a gait is, we define the *efficiency* of a gait as the normalized distance in the direction of the objective covered by the robot in a fixed simulation time, weighted by the distance that its trajectory is separated from a straight line. That is, we consider that a gait is better if the robot goes straight to the cone without any lateral deviation. We must point out that this measure is never used in the cognitive mechanism, it is just a way of presenting results in papers. In figure 5 we show the behaviour of this *efficiency* troughout the robot's life. As we can see, the curve tends to 1 as expected. Initially, the gaits are poor and the robot moves in irregular trajectories. This is reflected in the efficiency graph by the large variations

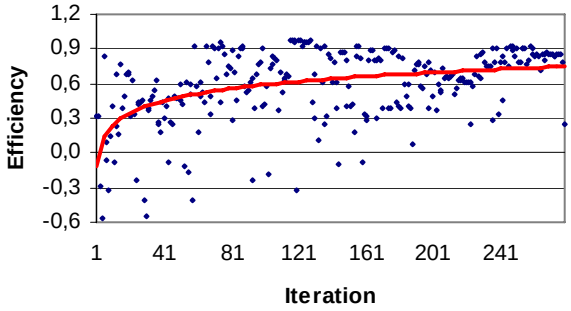

Fig. 5. Efficiency of the gaits applied by the robot. As time progresses it tends to one, this is, the robot moves straight to the cone without any lateral deviation.

in the efficiency from one instant to the next. Sometimes, by chance it reaches the cone, others it ends up very far away from it. As the interaction progresses, the robot learns to reach the cone without any deviation in a consistent manner and, the efficiency becomes generally close to one.
In the bottom image of figure 6, we can see the initial phase of each leg labelled with its corresponding name according to the schema shown in the top image for iteration 6 of the robot's life. The combination of phases is not appropriate for walking in a straight line and the robot turns, so the efficiency is low. In figure 7 we can see the gait applied in iteration 300. We display two images of the real robot and the simulation model applying this gait and a graphic representation of

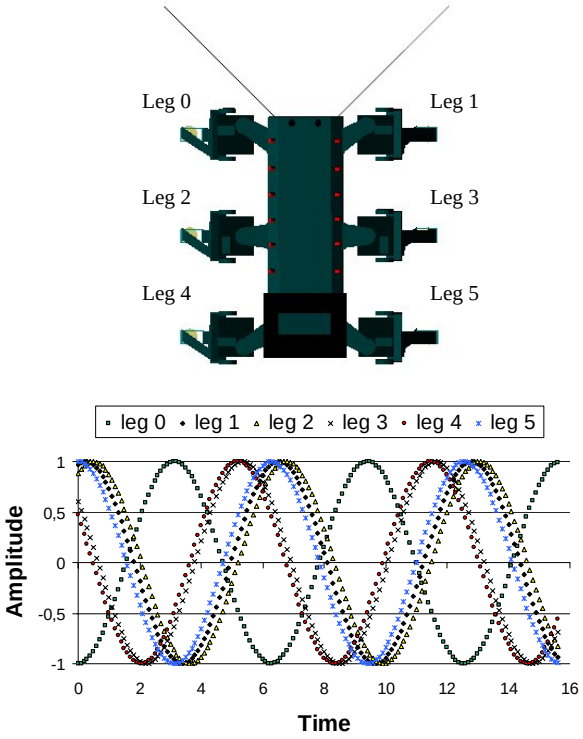

Fig. 6. The top figure displays the labeling of the legs, and the bottom figure displays the initial phase of the legs in iteration 6. As we can see, the combination of values seems to be random.

phases. As we can see, the initial phases are equal in groups of three and the resulting gait is quite good. In fact, the combination of phases leads to a very common and efficient gait called *tripod* gait (see [13]), where three legs move in phase and the other three legs in counter-phase resulting in a very fast and stable straight line motion. We must point out that this gait was developed by the robot itself, we just built the world for the robot to learn.

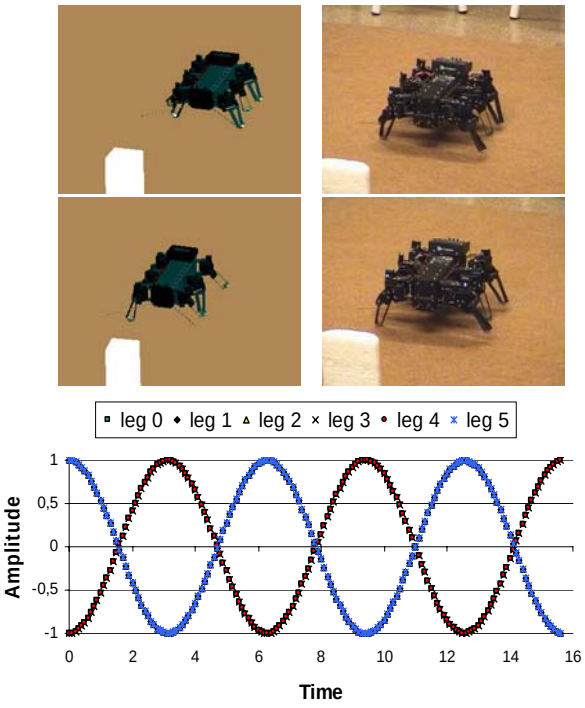

Fig. 7. The top figure displays 2 sequences of the applied gait in iteration 300, taken from the simulator in the left images and from the real robot in the right images. The bottom figure displays the initial phase of the legs in iteration 300 which leads to *a tripod* gait.

4.3. Robot learning to turn

At this point, the Hermes II robot has learnt to walk, and now we want it to learn to turn using the combination of initial phases obtained (*tripod* gait).

We place an object (a cone) in a semicircunference in front of the robot at a random distance between 50 and 100 cm, and the mechanism must select the best combination of amplitudes in the swing motion. The rest of parameters in the gait are fixed. If the robot reaches the cone (distance of less than 20 cm) or if it loses it (distance larger than 100 cm) we move it to a new position in the semicircunference. This way, we develop a teaching method as we would do with children: we present a objective and we reward the good actions.

The world model has three inputs, the distance and angle of the robot with respect to the cone (provided by the virtual sensor developed in [12]) and the amplitude of turn. The outputs are the predicted distance and angle. These two magnitudes are the inputs to the internal model, which has just one output, the internal state, which is maximum for the minimal distance and angle. Thus, the robot must reach the cone (minimizing distance) with low deviation (minimizing angle).

Both models are represented by artificial neural networks, whose parameters are evolved. The best results were obtained using two hidden layers of 4 neurons in the world models and two hidden layers of 3 neurons in the internal models. The population in the genetic algorithms was 600 individual for the world models and 300 for the internal models. As in the previous example, figure 8 shows the evolution of the error between the output provided by the world model and the real one.

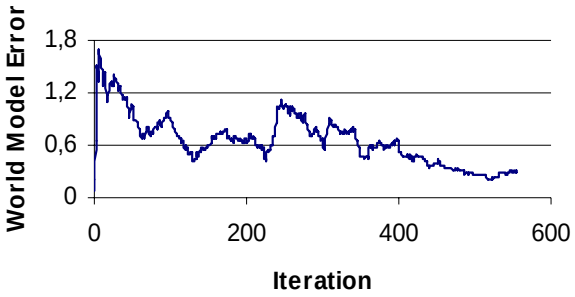

Fig. 8. Evolution of the mean squared error between the predicted distance and angle and the real values for the world models

In figure 9 we display the path followed by the robot with the strategies applied in iterations 53, 54, 55, 56 and 57 both for the simulation model and for the real robot. As we can see, in the first stages of the

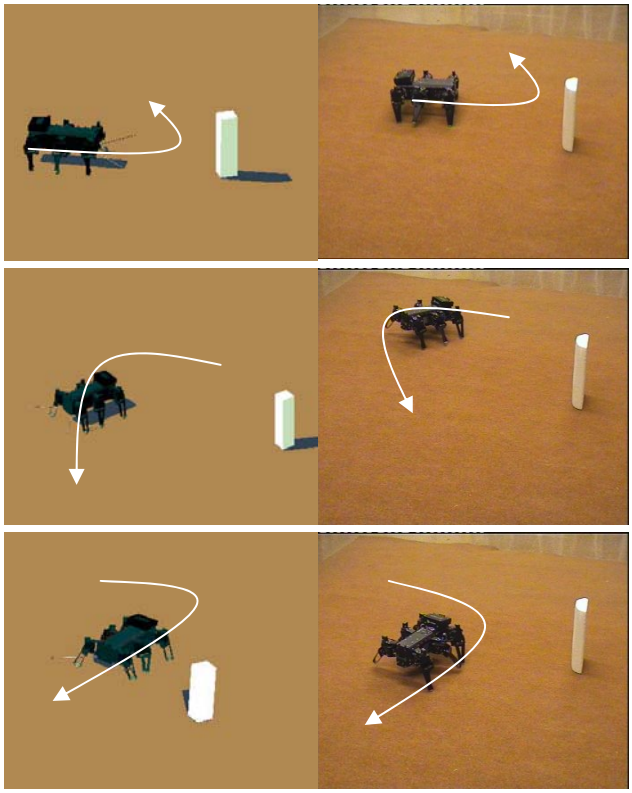

Fig. 9. Path followed by the robot in the simulator (left) and in the real robot (right) applying strategies 53, 54, 55, 56 and 57.

mechanism, the models are poor and the selected strategies are wrong. In fact, the cone remains in the same position during the application of these six strategies. After applying strategy 57, the robot loses the cone and we change its position.
In figure 10 we display the path followed in iterations 421, 422, 423, 424. The robot reaches the cone in iterations 422, 423 and 424 and we move it (in each image the cone is in a different position). The strategies are now very good and the robot is able to follow the cone turning in an adequate manner.

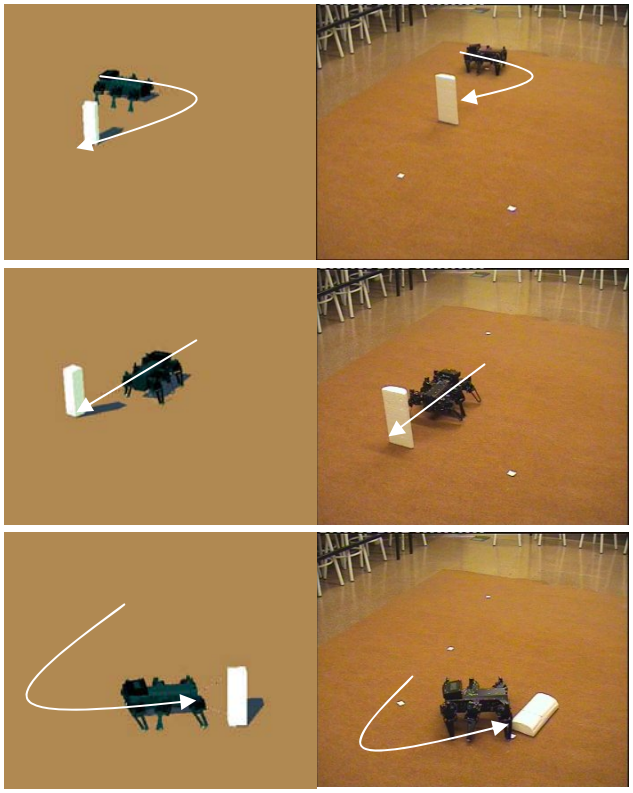

Fig. 10. Path followed by the robot in the simulator (left) and in the real robot (right) applying strategies 421, 422, 423 and 424.

5. Conclusions

We have presented a new cognitive mechanism that applies evolutionary techniques in order to supply an artificial agent with the capability of learning from its perceptions.
The mechanism was tested using a hexapod robot which was trying to learn to walk and to reach an objective. The results obtained are very promising, as the robot was able to autonomously obtain a tripod gait and modulate the amplitudes of the legs in order to reach an objective through continous interaction with the environment using its own sensors and internal motivations. This is very important because the mechanism permits the robot to find *the best solution* according to the limitations of its environment and its sensorial and actuation apparatus allowing it to adapt and survive in this particular world. One of the main features of this type of mechanisms is that if the world changes, the robot will adapt smoothly.
We have worked with a simulation model of the real robot that results very realistic, and the translation of the simulated behaviours to the real robot was perfect.
We are now working on the application of this type of mechanisms to robots that perform behaviours and need to learn tasks involving a higher level of complexity and behavioural diversity.

Acknowledgements

This work was supported by the FEDER through project PROTECAS N. 1FD1997-1262 MAR. and by the MCYT of Spain through project TIC2000-0739C0404.

References

[1] F. Prieto, R. J. Duro, J. Santos, Modelo de Mecanismo Mental Darwinista para robots móviles autónomos, *Proceedings of SAAEI'96*. Zaragoza, Spain, 1996.
[2] R. J. Duro, J. Santos, F. Bellas, A. Lamas, On Line Darwinist Cognitive Mechanism for an Artificial Organism, *Proceeding Supplement Book SAB2000*. pp 215-224.
[3] F. Bellas, J.A. Becerra, R. J. Duro, Using evolution for thinking and deciding, *Advances in Fuzzy Systems and Evolutionary Computation. pp 242-247.*
[4] M. Conrad, Evolutionary Learning Circuits, *J. Theor. Biol. 46*, 1974, pp. 167.
[5] M. Conrad, Complementary Molecular Models of Learning and Memory, *BioSystems 8*, 1976, pp. 119-138.
[6] J.P. Changeux, P. Courrege and A. Danchin, A Theory of the Epigenesis of Neural Networks by Selective Stabilization of Sinapsis, *Proc. Natl. Acad. Sci.* USA, 70, 1973, pp. 2974-2978.
[7] J.P. Changeux and A. Danchin, Selective Stabilization of Developing Synapsis as a Mechanism for the Specification of Neural Networks, *Nature 264*, 1976, pp. 705-712.
[8] J.P. Changeux, T. Heidmann., and P. Patte, Learning by Selection. *The Biology of Learning*, P. Marler and H.S. Terrace (Eds.), Berlin, 1984, pp. 115-133.
[9] G.M. Edelman, Neural Darwinism. The Theory of Neuronal Group Selection. *Basic Books* 1987.
[10] P. Nordin, W. Banzhaf and M. Brameier., Evolution of a World Model for a Miniature Robot Using Genetic Programming, *Robotics and Autonomous Systems, Vol. 25*, 1998, pp. 105-116.
[11] L. Steels, Emergent Functionality in Robotic Agents through On-line Evolution, *Artificial Life IV*, 1995, pp. 8-14 .
[12] Bellas F. , Becerra J.A., Santos J. and Duro R.J. Applying Synaptic Delays for Virtual Sensing and Actuation in Mobile Robots *Proceedings IJCNN 2000.*
[13] Beer R. D., Quinn R. D., Chiel H.J., Ritzmann R.E. Biologically Inspired Approaches to Robotics *Communications of the ACM, V.40 N. 3* pp. 31-39.