View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by El Servicio de Difusión de la Creación Intele

AN EXPERIMENTAL STUDY ON EVOLUTIONARY REACTIVE BEHAVIORS FOR MOBILE ROBOTS NAVIGATION

José A. Fernández León[1-3] Marcelo Tosini[1] Gerardo G. Acosta[2-3] H. N. Acosta[1]

[1] INTIA Research Institute – Computer and Systems Department
Exact Sciences Faculty - Universidad Nacional del Centro de la Provincia de Buenos Aires
(7000) Tandil - Buenos Aires – Argentina

[2] INTELYMEC Group – Electromechanical Engineering Department
Engineering Faculty - Universidad Nacional del Centro de la Provincia de Buenos Aires
(B7400JWI) Olavarría - Buenos Aires – Argentina

[3] CONICET - National Council of Scientific and Technological Research, Argentina

{jleon, mtosini, nacosta} @ exa.unicen.edu.ar gerardo.acosta @ ieee.org

ABSTRACT

Mobile robot's navigation and obstacle avoidance in an unknown and static environment is analyzed in this paper. From the guidance of position sensors, artificial neural network (ANN) based controllers settle the desired trajectory between current and a target point. Evolutionary algorithms were used to choose the best controller. This approach, known as Evolutionary Robotics (ER), commonly resorts to very simple ANN architectures. Although they include temporal processing, most of them do not consider the learned experience in the controller's evolution. Thus, the ER research presented in this article, focuses on the specification and testing of the ANN based controllers implemented when genetic mutations are performed from one generation to another. Discrete-Time Recurrent Neural Networks based controllers were tested, with two variants: plastic neural networks (PNN) and standard feed-forward (FFNN) networks. Also the way in which evolution was performed was also analyzed. As a result, controlled mutation do not exhibit major advantages against over the non controlled one, showing that diversity is more powerful than controlled adaptation.

Keywords: Evolutionary Robotics; Evolutionary Neural Networks; Robotic Adaptability; Simulated Robotic Agents

1. Introduction

According to Hebb [1], behavior is the primarily adaptation to the environment under sensory guidance. It takes the organism away from harmful events and toward favorable ones, or introduces changes in the immediate environment that make survival more likely. In this line of reasoning, Cliff et al. [9] and Nolfi [10] indicate that the most straightforward way to shape adaptive behavior is to use the evolutionary robotic (ER) approach. In ER behaviors are developed in close interaction with the environment, limiting the human intervention to set a target behavior. Then, a rule for determining how much a given behavior approximates to the one desired must be specified.

In this paper, ER concepts are used to obtain controllers that can adapt the robot's behavior according to its sensory inputs.

1.1. Evolutionary Robotics

Evolutionary Robotics (ER) is a sub-field of Behavioral Robotics and it is concerned with the application of evolutionary computation methods to the area of autonomous robotic control systems. One of the central goals of ER is to develop automated methods that can be used to evolve complex behavior-based control strategies [2].

Several works pointed out that one of the main interests in ER is the development of computational intelligence based control systems or intelligent control systems for short [2][3][4][5][6][7]. Mainly, these works describe experimental proof about obstacle avoidance, maze exploration, robot learning, and adaptive controllers using small mobile robots or computational models of them (simulated approach).

1.2. Artificial Agents

According to Russell and Norvig [8], "an agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors". In addition, in [8] an "ideal agent" is defined as "one that always takes the action that is expected to maximize its performance measure, given the perceived sequence it has seen so far". Based on it, an agent that maximizes its fitness (expressed by a mathematical fitness function) will be considered appropriated for a certain task in a specific environment. In the present context, to construct an agent that adapts

himself in an appropriate way retaining and responding to information from the environment are one of the goals to overcome.

1.3. Evolutionary Robotics and Adaptation in Artificial Systems

In a natural system, animals can be considered as falling into the category of one of these agents. In fact, animals do not only adapt to environmental changes, but they can also accumulate adaptations. They can store "knowledge" about a previously encountered environment and use it to alter their behavior when faced with a specific environment again. This process is called *learning* when it occurs in a lifetime and *evolution* when it occurs in a lineage [13]. In other words, the agent adapts itself by evolving different behaviors appropriate for different environments. However, it is important to notice that the definition of appropriate for each environment is defined explicitly without ambiguity in a fitness function. In addition, in [13] it is not considered directly the interaction between evolution and learning, but they make the assumption that all previous generations have been exposed to the same set of environments. This assumption is a bit strong and unrealistic. Instead, it can be assumed that the agent had been able to accumulate adaptations with a slow change on the experience obtained in one generation. In this particular instance, the experience obtained by a neuro-controller refers to the features that it might develop in a generation (i.e., the synaptic weights established after an evolution process).

One of the main challenges in ER is to discover and to model different adaptation mechanisms. Most of the works in ER [3][10][11] consider the artificial evolution of neuro-controllers as one of these adaptation mechanisms. It is considered a viable methodology to develop autonomous agents that could exhibit conscious abilities. Artificial evolution differs from other learning schemes because it works on a population of different individuals and it is based on a selectionist approach, rather than a goal-directed one [12]. This is the main approach adopted in this work.

1.4. Artificial Neural Networks in Evolutionary Robotics

Artificial Neural Networks (ANN) have become the chosen computational structure in ER. ANN based controllers have been implemented for different functions in mobile robots [2][4][14][15]. Most of these applications developed simple ANN architectures, which are capable of temporal processing. Typical examples are the Discrete-Time Recurrent Neural Networks (TRNN) with two variants: Plastic Neural Networks (PNN) used in [16][17], and a variant of Feed-Forward (FFNN) described in [18][19]. This kind of controllers is capable of behaving properly, remembering the acquired abilities and passing it to the next generations. They can store previous experience and use it to alter their present behavior, as well as their descendants' behavior, when faced with an environment (or situation). This adaptation is faster as the fitness function is achieved.

In this paper, an evolutionary robot control system is examined in a simulated environment through generation of neuro-controllers in an artificial evolutionary process. Description of the simulation environment, the implemented neuro-controllers and their evolution through the use of genetic algorithms, is given in the following sections. The article also includes results and preliminary conclusions on the use of ANN based controllers in ER, advantages and disadvantages of permanent adaptation, and the influence of the speed of adaptation in the general behavior.

2. Evolutionary neuro-controllers and simulation environment

2.1. Robot description

The robot used was Khepera® [25] which provides a simple model of mobile robot that is frequently used in ER area.

The inputs to the neuro-controllers consist of the readings of three distance sensors separated 45° one from another in the frontal half of the robot and four light sensors. Distance sensor outputs a positive value if there is an obstacle in its direction and within 15 length units of the robots. Light sensor's output value is negatively correlated to the angle between the sensor and the source, and also to the distance between robot an source. Fig. 1 shows the sensors positions according to robot model.

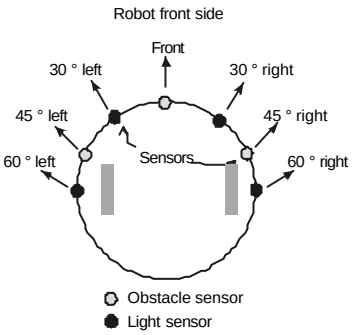


Fig. 1 - Sensory layout

2.2. The artificial neuro-controllers

The ANN based controllers used in the simulated environment respond to the agent paradigm described previously. The ANN selected for testing correspond to Discrete-Time Neural Networks in the way of PNN and FFNN networks.

Two classes of tests were developed. The first

was related with non-recurrent networks (PNN and FFNN) and the second one adding recurrence at hidden level of the previous ones. The FFNN non-recurrent network used is a simple network similar to Braintenberg vehicle [21] in that they produce the output signal in direct response to current range sensor readings.

The single-layered recurrent networks have the possibility of developing temporal processing. The recurrent connections allow ANN to remember the action taken at a previous temporal stage. According to [2], controllers that can make use of temporal information have the potential to outperform completely reactive controllers considering simple sensors used by the robots.

In evolutionary terms, only the weights of FFNN networks were evolved, instead of other parameters were evolved for PNN. These parameters are signaled in section 2.3, in the same way as [18].

To implement the ANN based controller, two neurons with sigmoid activation function were used in the hidden layer. The output layer consisted of one neuron with sigmoid activation function.

For FFNN networks several instances were analyzed:

- Single FFNN without hidden-layer recurrence vs. FFNN with hidden-layer recurrence
- Random weights initialization (RANDOM-INIT) vs. apriori weight-sign initialization (PRESEL-INIT) for FFNN with recurrence
- Randomized weight mutation (TOTAL-MUTATION) vs. controlled mutation (PARTIAL-MUTATION) for FFNN with recurrence.

For recurrent PNN, a priori vs. randomized sign initialization was analyzed.

The typical network configuration used is shown in Fig. 2. However setting recurrent weights in the hidden layer to zero (dot arrows) allows to remove recurrence connection to implement non-recurrent networks.

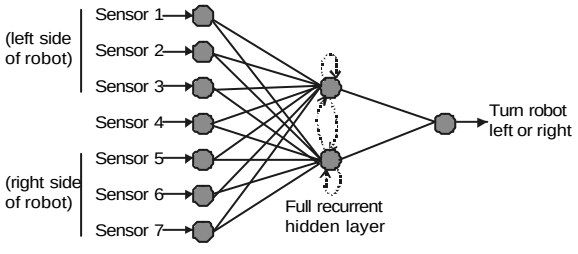


Fig. 2 - The neural network topology. The input layer consists of seven receptors fully connected to two hidden neurons. A set of recurrent connections are added to the hidden units only for the recurrent networks implemented. The hidden units were not connected to two motor-actuators, which are connected to an output neuron modeling the deviation robot angle.

2.3. The simulation environment

The simulation environment consists of planar objects in which each component is a solid occupying a space in this environment. The number of objects is fixed to 10, and their position is randomized. During each generation, the members of the population are initialized to also random locations in the environment space. This is a continuous region in R^2 allowing the simulation of the robot's movements. Also, there is one light source in a random location. This light represents the goal position to be reached by the robot.

In each environment, just one robot, which is specified as a structure that maintain the robots current position, number of sensors, and sensor input readings, is simulated. In addition, both the neuro-controller structure and the evolved algorithms are associated with the robot.

It is important to remark that the robotic-agent model does not consider actuators like motors and wheels: only the robot deviation angle is considered in each simulation step. This ideal (supervisory control level) approach allows the robot to turn in any direction or go straightforward.

2.4. The Simulation Environment

To carry out the tests, two simulation environments were used: one for learning and the other for validation.

The learning environment was represented by two-dimensional rectangles (x, y). For experiments, ten objects were ramdomicaly placed inside the environment. In each generation, the population's members (different instances of neuro-controllers for the robot) were also initialized in random positions inside the environment. Besides the obstacles, the environment had a source of light in a random position for each run. This light represents the goal to be reached by the robot. Each environment that allows the simulation of a run was specified as a structure containing the robot and the objects position (obstacles and lights). Furthermore, both neuro-controller's structure and evolutionary algorithms are associated to robot's instance generated for the learning. Neither dynamic models nor non-linealities in actuator model (motors and wheels) were considered. Instead, in each simulation step the angle of the robot's deviation regarding their objective was taked into account.

When the learning stage is over, the best robot controller was selected and it was evaluated in a more complex simulator called YAKS [22]. The results obtained are qualitatively similar to those reached in the previous learning environment.

2.5. The evolutionary algorithm

Each neuro-controller was specified using an evolutionary computing algorithm and tested into

the environment. For PNN controller, the Hebbian rules were determined according to [7][18]. While for FFNN networks connection weights were initialized with random values in [-2, 2] range, for RANDOM-INIT configuration, or random values in [0, 2] range with preselected sign, for PRESEL-INIT setting.

The chromosome data structure (Table 1) represent a controller like the one specified in [18] for *genetically determine*d controllers (e.g. FFNN) (a sign and weight strength for each synapses), and for *adaptive synapse controllers* (e.g. PNN) (a sign, a specific Hebb-adaptive rule, and a learning rate).

Genotype encoding	**Values for one synapse**		
FFNN	sign	weight	
PNN	sign	Hebb rule	rate

Table 1 - Genetic encoding for synaptic parameters. For FFNN controllers, a signed weight for each synapse. For PNN controllers, a weight-sign, one (of four) Hebbian rules and a mutation rate [18].

According to [20], the adaptation ignores the diversity. Referring to neuro-controllers' mutations in an evolutionary scale, diversity is related to a free random selection of genetic values. Instead the mutation criteria refers to a controlled random selection of genetic mutation parameters (e.g. synaptic weight mutations and its mutation rates). In this work, weights are mutated with a fixed mutation rate of 50%.

The mutation criterion selected for this work is as follows. Each weight magnitude for a synaptic connection in a FFNN network (genetically determined controller) depends on the accumulated adaptation by the evolutionary process [11], and it is affected by a randomized adaptation rate in the [-2, 2] range, for TOTAL-MUTATION configuration, or a variation in [-0.25, 0.25] range over original mutated weight, for PARTIAL-MUTATION setting, valid only for FFNN. This small adaptation rate slowly provokes mutations to the neuro-controller in an evolutionary scale.

For FFNN case, weights (w) were mutated using the following equation:

$$w = w + u * R \qquad (1)$$

where u is the mutation rate in [0, 1] range and R is a randomized value in [-0.25, 0.25] range. The effect of the weight mutation Eq. (1) is a slow drift from one generation to another. Instead, for PNN the mutation refers only to sign changes, while variations in magnitude weight are determined by Hebbian rules into the evaluation process.

This mutation criterion permits a neuro-controller with good fitness to be close to several descendants in the next generation in a genetic mutation. As a result of this, descendants will be also appropriately adapted to the environment, taking advantage of the acquired experience.

2.6. The fitness function

The performance evaluation for each controller is based on a variant of the performance fitness function showed in [2]. The net offset between a robot starting position and its final position, and whether or not the robot becomes stuck within the simulated environment are considered to write down the fitness function. It is shown in equation (2).

$$F(c_i) = \begin{cases} F(c_i) + 1 & \text{if } k_1 \text{ Ù } k_2 \\ F(c_i) & \text{else} \end{cases} \qquad (2)$$

where k_1 relates the blocking of the robot near an obstacle. Therefore, the controllers that are most times blocked will obtain a lower fitness level than others presenting an avoidance obstacle behavior; and k_2 states the offset between a robot's starting and final position. The refered fitness level is normalized between the possible successful actions during the controller life time (iterations). Fig. 3 shows the fitness function evolution in the [0; 1] range.

A robot that cannot avoid objects will soon became immobilized when its path is blocked, obtaining the robot controller a low fitness result.

3. Results and discussion

Obstacle avoidance and navigation behaviors are used in several tests in the ER literature [3]. They consist of evolutionary training of specific neural controllers to obstacles avoidance or towards navigating to a specific point within enclosed areas [2][19].

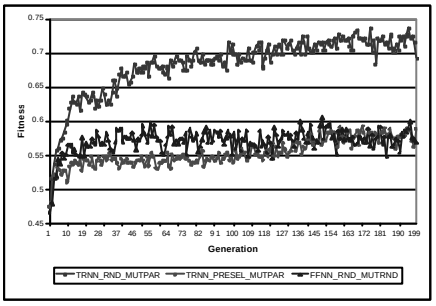
In our previous tests (e.g., [23][24]) and in [2] it was found that single-hidden layer feed-forward neural networks were capable to control robotic-agents in a simulated environment, similar to the one used in this work. It was possible when the robot sensors were simple approximation sensor models. A simple Braitenberg vehicle was implemented to effectively perform the obstacle avoidance task reasonably well in the simulated environment. The Braitenber's controller was selected because it had no necessity of reading information fron the past to overcome perceptual aliasing. These controllers had no capacity for temporal processing.

Seven alternatives of FFNN and PNN networks were trained with 30 genotypes (20 chromosome each one) over 200 generations and 10 runs for each neuro-controller (or phenotype). Evolved controllers show different behaviors depending on the initialization and mutation characteristics. Particularly, neuro-controllers with pre-selected

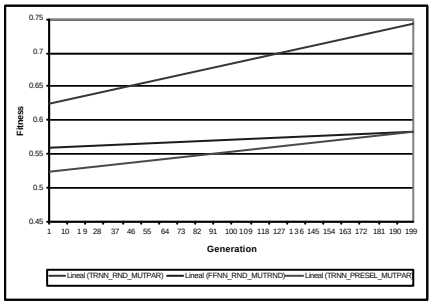
weight-sign and/or controlled weight mutation evolved better than others with random initialization and mutation. It reffers to controllers with shortest navigational paths. The robotic-agent in each test signaled was evolved with a standard mutation criteria for PNN and a generational mutation criteria for FFNN.

It was found that TRNN_PRESEL_MUTPAR configuration on FFNN produces the best results in least amount of simulation time (or generations). Networks settings described in section 2.1 were tested with fixed synaptic weights, and with randomized weights (Fig. 3). According to performed test controllers with preselected weight-sign and/or controlled weight mutation evolved better than others.

Fig. 3 shows the average of fitness evolution generated during a typical run for the tested recurrent FFNN configurations. At each generation the best, the worst, and the average controller behavior performances were recorded.



(a)



(b)

Fig. 3 – Results from three evolutionary generations. In this graph, (a) the fitness function is plotted against generation for best (TRNN_PRESEL_MUTPAR), medium (FFNN_RND_MUTRND), and worst (TRNN_RND_MUTPAR) studied controllers; (b) lineal tendency for best, medium, and worst controllers. Evolved controllers shows different behaviors depending on the initialization and mutation characteristics.

Fig. 4 shows the average of fitness evolution generated during a typical run for the tested PNN configurations (with total and controlled weight-sign mutation) and recurrent FFNN with a priori weight-sign initialization and controlled weight mutation (best controller of Fig. 3). Training performance was averaged for each controller for one simulation of 200 steps each one with selection and mutation processes.

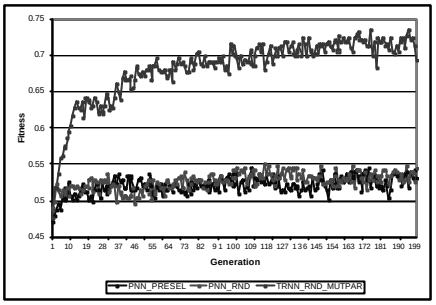
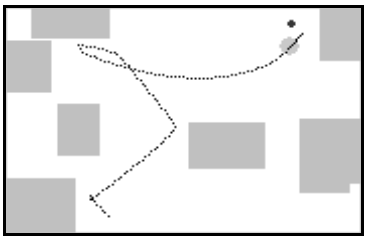


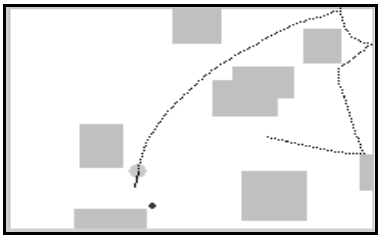
Fig. 4 - Comparative fitness evolution for the best controller of figure 3 (TRNN_PRESEL_MUTPAR) and two alternative PNN controllers (with total and controlled weight-sign mutation).

The results of the mutation criteria selected in this work (section 2.3) shows that the controlled mutation does not present advantages over those with non-controlled mutations. It means that diversity is more powerful than controlled adaptations.

Simulation shows that the robotic-agent is able to avoid obstacle with recurrent configuration after encountering an obstacle and backing out of sensor range. Robotic-agent also displayed movement sequences appreciated before in time. This indicates that the neuro-controller with evolving capabilities proposed in this work resort to previous responses as expected (Fig. 5.b). In addition, it was appreciated the robot lost the light source when it passed near this point, because the robot light-sensors did not receive these stimules.



(a)



(b)

Fig. 5. An example of simulated movement of the robot with neuro-controller for obstacle avoidance. The lines indicate the paths taken by the robot during the curse of the simulation, and the red point represents the light source or robot goal point. Figure 5. a - for non-recurrent FFNN network and Figure 5. b - for recurrent PNN network, both of them with no weight-sign initialization (random initialization).

4. Conclusions and future work

The work presented here describe in general terms robot controllers instantiated in simulated obstacle avoidance environment. Different neuro-controllers were evolved in simulation and the best of them was selected in each generation to obtain an appropriate final controller. The robot model was developed based on evolutionary and adaptive criteria.

Some recurrent neuro-controllers showed special behavior: they have a tendency to do circular movements after avoiding an obstacle. Also, a more detailed study of the sensors layout should be developed to avoid erratic behaviors when the robot finds the goal.

This work demonstrates once more the feasibility in application of ANN based controllers on ER, showing its potentials as regards as adaptability and learning behaviors.

Future work will be related to obtain neuro-controllers with architectures like the one presented here or similar, for mobile robots in real physical environments.

Acknowledgments

To the R+D network RIDIAAR (*Red de Investigación y Desarrollo en Inteligencia Artificial Aplicada a Robótica*), conformed by the INTELYME e INTIA Groups, supported by the UNICEN.

This research was also partially supported by a Marie-Curie International Fellowship within the 6th European Community Framework Programme (Project 03027 – AUVI).

5. References

[1] D. O. Hebb. A Textbook of Psychology. Philadelphia, PA., W. B. Saunders, 1958, pp 44-45.

[2] A. L. Nelson; E. Grant; J. M. Galeotti; S. Rhody. Maze exploration behaviors using an integrated evolutionary robotic environment. Robotic and Autonomous Systems 46. 2004, pp. 159-173.

[3] Nolfi S. and Floreano, D. Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines. MA: MIT Press/Bradford Books. 2000.

[4] Elio Tuci, Inman Harvey, and Matt Quinn. Evolving integrated controllers for autonomous learning robots using dynamic neural networks. Proceedings of The Seventh International Conference on the Simulation of Adaptive Behavior (SAP'02), 49 August 2002, Edimburgh, UK.

[5] Lipson, H. Uncontrolled Engineering: A review of Nolfi and Floreano's Evolutionary Robotics. 2000.

[6] Urzelai, J. and Floreano, D. Evolution of adaptive synapses: Robots with fast adaptive behavior in new environments. Evolutionary Computation, 9:495-524. 2001.

[7] E. Tuci, M. Quinn. Behavioural plasticity in autonomous agents: a comparison between two types of controller. Proceedings of The Second European Workshop on Evolutionary Robotics EvoROB2003, 14-16 April 2003, Essex, UK, pp. 661-672.

[8] S. Russell and P. Norvig. Artificial Intelligence: A Modern Approach, Prentice-Hall, Inc. 1995.

[9] D. Cliff; I. Harvey, and P. Husbands. Explorations in Evolutionary Robotics. Adaptive Behavior, 2:73-110. 1993.

[10] Nolfi, S. Adaptation as a more powerful than decomposition and integration: Experimental evidences from evolutionary robotics. In P. K. Simpson (Ed.), Proceedings of the IEEE International Conference on Fuzzy Systems (FUZZ-IEEE'98), New York: IEEE Press, 141-146. 1998.

[11] D. Floreano. Ago Ergo Sum. In Mulhauser, G. Editor, Evolving Consciousness. Benjamins Press, New York. 1997.

[12] L. Steels. Building agents out of autonomous behavior systems. In L. Steels and R. Brooks, editors. The Artificial Life route to Artificial Intelligence building situated embodied agents, pp. 102-137. Lawrence Erlbaum, New Haven, 1993.

[13] C. Fernando. Accumulation of Adaptations in Plastic Neural Networks. MSc. Dissertation. COGS, University of Sussex. 2002.

[14] D. Floreano and F. Mondada. Evolutionary Neurocontrollers for autonomous Mobile Robots. Neural Networks, 11:1461-1478, 1998.

[15] J. A. Driscoll, R. A. Peters II. A development environment for evolutionary robotics. Proceedings of the 2000 IEEE International Conference on Systems, Man, and Cybernetics, vol., pp. 3841-3845. 2000.

[16] D. Floreano and J. Urzelai. Neural morphogenesis, synaptic plasticity, and evolution. Theory in Biosciences, 120 (3-4), 223-238. 2001.

[17] S. Nolfi and D. Floreano. Learning and evolution. Autonomous Robots, 7(1): 89-113, 1999.

[18] Floreano, D. and Urzelai, J. (1999) Evolution of Adaptive-Synapse Controllers. In D. Floreano et al. (Eds.), Advances in Artificial Life. Proceedings of the 5th European Conference on Artificial Life, Berlin: Springer Verlag. (ECAL'1999). 1999.

[19] Togelius, J. Evolution of th Layers in a Subsumption Architecture Robot Controller. Master of Science in Evolutionary and Adaptive Systems. University of Sussex, UK. 2003.

[20] D. E. Goldberg. Genetic Algorithms in search, optimization, and machine learning. Addison-Wesley. 1989.

[21] Braitenberg, V. Vehicles: Experiments in synthetic psychology. Cambridge, MA: MIT Press. 1984.

[22] Karlson, J. YAKS Yet Another Khepera Simulator. URL: http://r2d2.ida.his.se/ . 2002.

[23] Fernández León, J. A.; Tosini, M.; Acosta, G. G. Evolutionary Reactive Behavior for Mobile Robots Navigation. IEEE Conference on Cybernetics and Intelligent Systems (CIS). Proceedings of the 2004 IEEE CIS, Singapore, pp. 532-537. 2004.

[24] Fernández León, J. A.; Acosta, G. G.; Mayosky, Miguel A. Estudio de Neuro-Controladores Evolutivos para Navegación de Robots Autónomos. Maestría en Ingeniería de Sistemas. UNCPBA, Argentina.2005.

[25] Khepera, mini robot. K-Team. http://www.k-team.com/robots/khepera /index.html 2004.