Acquiring Adaptive Behaviors of Mobile Robots Using Genetic Algorithms and Artificial Neural Networks

Nicolás Navarro[1], Cesar Muñoz[1], Wolfgang Freund[2], Tomás Arredondo V.[2]

[1]*Centro de Robótica*
e-mail: centro.robotica@gmail.com
[2]*Departamento de Electrónica*
Universidad Técnica Federico Santa María
Casilla 110V, Valparaíso, Chile.

Abstract

This paper describes the use of soft computing techniques for acquiring adaptive behaviors to be used in mobile robot exploration. Action-based Environment Modeling (AEM) based navigation is used within unknown environments and unsupervised adaptive learning is used for obtaining of the dynamic behaviors. In this investigation it is shown that this unsupervised adaptive method is capable of training a simple low cost robot towards developing highly fit behaviors within a diverse set of complex environments.

The experiments that endorse these affirmations were made in Khepera robot simulator. The robot makes use of a neural network to interpret the measurements from the robot sensors in order to determine its next behavior. The training of this network was made using a Genetic Algorithm (GA), where each individual robot is constituted by a neural network. Fitness evaluation provides the quality of robot behavior with respect to his exploration capability within his environment.

1. Introduction

During the last decades numerous and extensive investigations have been made using soft computing techniques towards the problem of navigation in diverse environments by mobile robots. The method used in this paper, consists of providing a neural network brain to the robot that allows it to determine its motion based on sensor inputs. The inputs of this network are the values measured by infrared sensors which give limited information about the surroundings in which the robot is located. The resulting network output indicates the direction of rotation for two DC motors that allow the displacement of the robot. The robot movement is composed of a sequence of four basic actions. These actions come from the use of AEM [1, 2]. For training this network a GA is utilized. The GA selects from a population of robots (neural networks) using a fitness function [4, 7] that privileges the capacity of exploration of the robot. That is, to cross the greatest amount of previously unknown space within a limited amount of actions. This method is chosen due to the difficulty in programming specific strategies to follow for the diverse conditions and environments which the robot can be faced with [6]. This type of method can be used in the construction and/or update of maps based on limited sensor information for tasks like cleaning floors as is done by the ROOMBA robot [9, 10].

The organization of this paper is as follows. In section 2 the robotic simulation system is provided. In section 3 the experiments performed are given. The results analysis is given in section 4. Finally, in section 5 some conclusions and future work are given.

2. Robotics Simulation System Description

This section presents the main features of the robotic simulation system used for these studies. The simulation system has several different elements including: the robot simulator [3], AEM [2], neural networks, GA.

2.1. Khepera Robot

For these simulations, a Khepera robot was chosen (Figure 1). The robot configuration as tested has two DC motors and eight sensors of infrared proximity with which it is possible to detect objects up to a short distance. These low cost sensors provide the robot only with local data around itself. The robot does not have global position information. The infrared sensors provide 10 bit output values, which allow the robot to know in approximate form the distance to obstacles. In order to make the experiments as close to reality as possible a 5% random noise was introduced to the readings. A zero value indicates no obstacle is found, and a one indicates that the robot is close or in contact with an obstacle. In Figure 2 the distribution of the sensors can be appreciated.

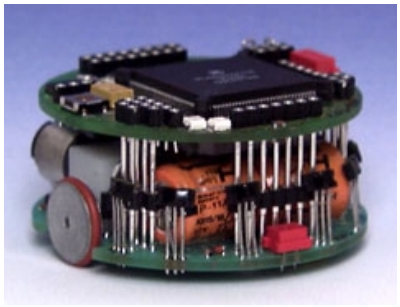

Figure 1. Khepera.

2.2. Simulation System and AEM

These experiments were made using the YAKS [3] Khepera simulator. The simulator has a map where the robot moves, the simulator provides the readings for the sensors according to the current map (room). It also handles the information of zones visited, not visited and the various obstacles in the room. The rooms are square with length and width of 2750 mm in size. The rooms are differentiated by the amount and type of obstacles they present to the robot. The rooms were divided into 2500 zones (each 55 mm by 55 mm). The robot generates an internal map in which the zones are marked with various values: obstacles are indicated with a value of -1, those not visited by the robot are marked with 0 and the visited ones with 1. The robot executes 1000 steps in each simulation, not every step produces forward motion as some only rotate the robot.

In order to reduce the search space of behaviors, we use a limited number of actions for the robot to execute in each step. Using a similar encoding as in the paper by Yamada [1], four basic actions were used:

- A1: Go 55 mm straight on.
- A2: Turn 30º left.
- A3: Turn 30º right.
- A4: Turn 180º left.

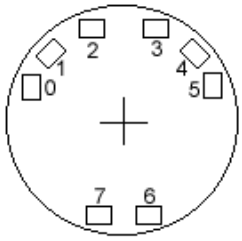

Figure 2. Distribution of Khepera sensors.

2.3. Applied Artificial Neural Network (ANN)

The neural network [7] used in this investigation uses: eight input neurons (one for each infrared sensor), five neurons in the hidden layer and two output neurons directly connected to the motors that produce the robot movement as shown in Figure 3.

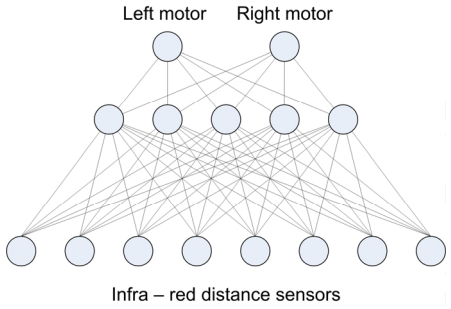

Figure 3. Configuration of the Neural Network.

2.4. Genetic Algorithm

As already mentioned a GA [1, 8] is used to find an optimal configuration of weights in the neural network [4, 7]. Each individual in the GA represents a neural network which is evolving with the passing of different generations.

The GA uses the following parameters:

- Population size: 200.
- Crossover operator: Random crossover.
- Selection method: Elite strategy selection.
- Mutation rate P_{mut}: 1%.
- Generations: 180.

The GA procedure is as follows:

Step 1: Initializing population: An initial population $I_1,...,I_n$ is randomly generated.

Step 2: Robot executes behavior: The behavior of each robot is simulated.

Step 3: Computing fitness: The fitness $f_1,...,f_n$ for each individual $I_1,...,I_n$ is computed based on complete behavior in the room.

Step 4: Selection: Using the fitness values $f_1,...,f_n$, select an elite group of ten (10) individuals from the current population (C).

Step 5: Crossover: For each elite individual select ten mates from the rest of the population for the next generation. The rest of the individuals are randomly generated, forming C_1.

Step 6: Mutation: Mutate the individuals in C_1 based on mutation rate P_{mut}.

Step 7: Go to Step 2.

In Step 5, we decided to generate 50% of the population at random to obtain a greater diversity of solutions and to limit the issue of early convergence.

2.5. Fitness function

The fitness function [5] uses the information provided by the simulator and the capacity of exploration of each individual as follows:

$$f_i = \frac{Z_v}{Z_{\max}}$$

Where:

f_i : is the fitness of an individual.

Z_v : visited zones.

$Z_{\max}$: maximum number of possible visited zones.

The interaction of the complete system can be seen in Figure 4. The GA selects the best individuals in each generation and modifies the weights based on the f_i fitness function after each individual executes his behavior in each room. The GA selects the overall best individual in the last generation of the simulation.

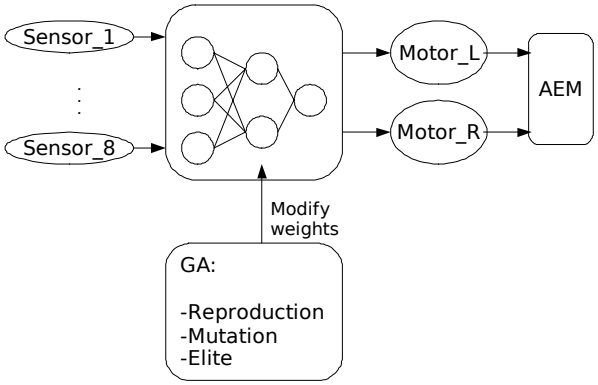

Figure 4. System Interaction

3. Experiments

The experiments performed included two cases and six different rooms:

The first case (case 1) evaluated consisted of consecutively training a population in three different rooms, that is, the first generation is created and is trained on room 1 until obtaining the generation number 60, after this generation, the population is trained on room 2 until generation 120. Finally the last 60 generations are trained on room 3. Next, the best individual of the 180 generations is chosen and tested in the 6 rooms. The exploration percentage of each room is used for fitness estimation

This procedure was repeated and tested for different rooms. This case was made with the purpose of presenting a greater set of environments with the hope that the robot would acquire more complex and robust behaviors.

The second case studied (case 2), consists of training a single population in a single room by 180 generations and taking the best individual from the last generation. This individual is also tested for it's exploration capacity in all 6 rooms. Finally a performance comparison of the different obtained behaviors is made, based on the exploration percentage of the rooms. In all experiments the starting point of exploration is in the lower left corner of the room.

3.1. Experimental Results

In Figures 5 – 10 we show some experimental results. In all these figures, straight lines represent walls and circles simulates obstacles.

In Figures 5 – 7, we show representative behavior results for case 1.

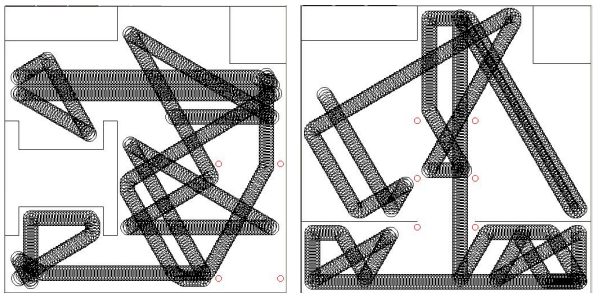

Figure 5. Exploration in Room 1 and 2 for case 1.

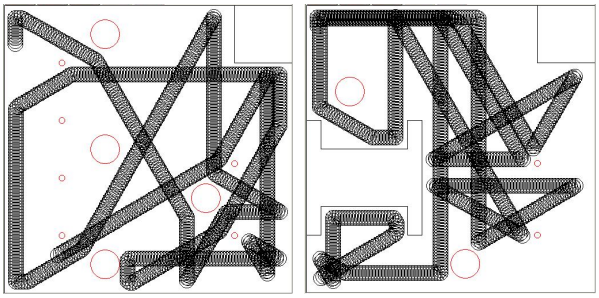

Figure 6. Exploration in Room 3 and 4 for case 1.

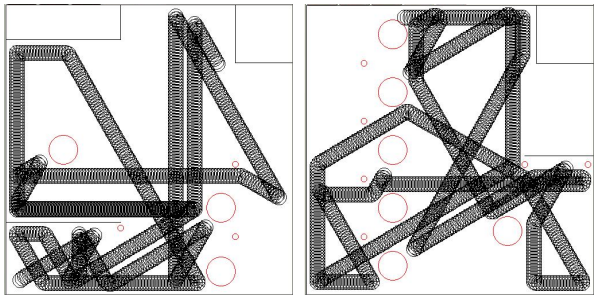

Figure 7. Exploration in Room 5 and 6 for case 1.

Figures 8 - 10, show representative behaviors obtained for case 2:

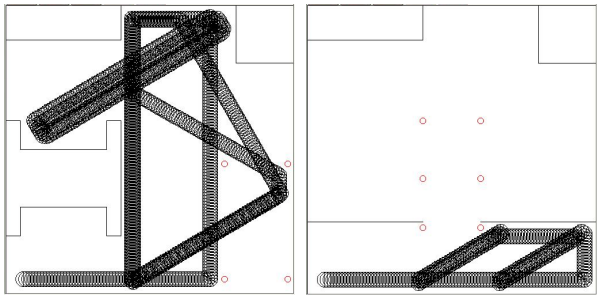

Figure 8. Exploration in Room 1 and 2 for case 2.

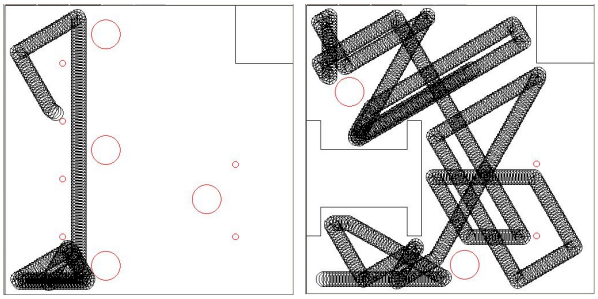

Figure 9. Exploration in Room 3 and 4 for case 2.

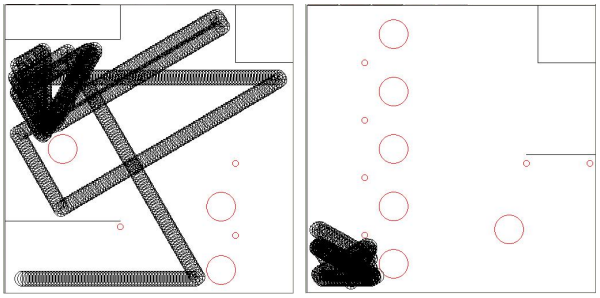

Figure 10. Exploration in Room 5 and 6 for case 2.

4. Analysis and Evaluation

Observing Figure 5-10, we can conclude that the robot behavior acquired in case 1, which was trained on multiple rooms, is much more effective, for exploration purposes, than that obtained in case 2, which was only trained on a single environment.

This result could be explained through a biological parallelism. It is known that individuals that are constantly intellectually stimulated, are able to solve more complex problems and in less time than those with poor stimulation. In the simulation, this phenomenon could be attributed to the use of genetic algorithms for optimization together with neural networks which when trained with variety of stimuli respond by interpolating solutions for unforeseen cases.

In Table 1and in Figures 11 - 14 we show a more quantitative analysis of our the previous analysis.

Table 1. Exploration % of the Best Individual

	Best Individual Case 1	Best Individual Case 2
Room 1	77,0	71,0
Room 2	82,5	65,0
Room 3	81,4	66,0
Room 4	75,0	71,0
Room 5	74,0	68,0
Room 6	69,0	29,0

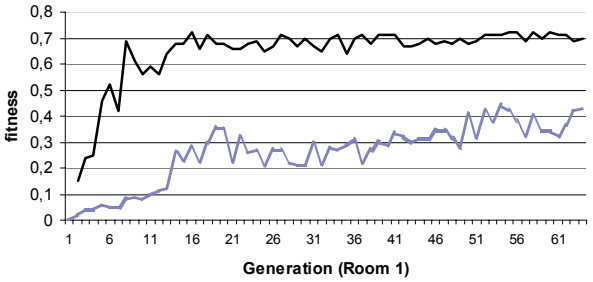

Figure 11. Fitness Evolution for Case 1, Generations 0 - 60.

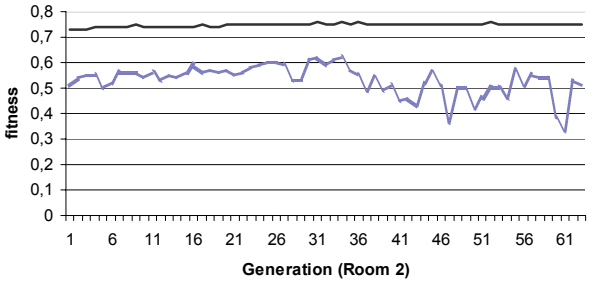

Figure 12. Fitness Evolution for Case 1, Generations 61-120.

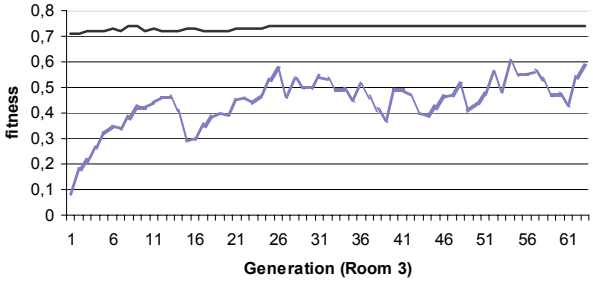

Figure 13. Fitness Evolution for Case 1, Generations 121-180.

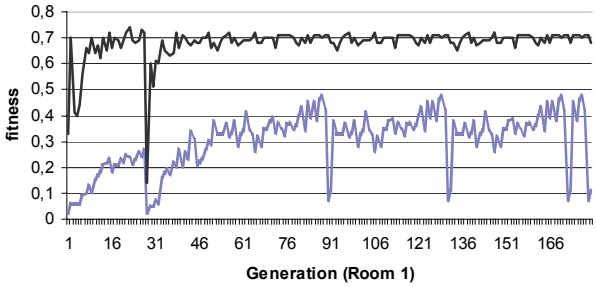

Figure 14. Fitness Evolution for Case 2, Generations 0-180.

5. Conclusions and Future Work

The results presented in this paper are promising, based on the wide range of applicability. We show that the robot behavior, which was trained on multiple rooms, is much more effective, for exploration purposes, than that obtained by using single environment training.

Following our experimental results, we conclude that using a low cost robot with simple sensors, (such as infrared), could be used in several different applications of navigation such as: minefield detection, radioactive contaminated area exploration, survivors search in catastrophe zones, exploring contaminated sites, and others. More common applications scenarios could include applications such as cleaning rugs like done by the Roomba robot [9].

Moreover, the obtained results motivate us in new areas of interest, such as validation of the results in a real robot and introducing new behavioral motivations other than exploration.

Finally, by incorporating new sensors to our robot we hope to obtain better knowledge of the environment to resolve more complex tasks.

6. Acknowledgements

We would like to thank Elena Villanueva, master student of Automation of the Universidad Técnica Federico Santa Maria and the Centro de Robótica of Universidad Técnica Federico Santa María.

7. References

[1] S. Yamada, "Evolutionary behavior learning for action-based environment modeling by a mobile robot", Applied Soft Computing 5 (2005) 245–257.

[2] S. Yamada, “Recognizing environments from action sequences using self-organizing maps”, Applied Soft Computing 4 (2004) 35–47.

[3] YAKS simulator website: http://r2d2.ida.his.se/

[4] Frank Hoffmann, “Soft computing techniques for the design of mobile robot behaviors”, Information Sciences 122 (2000) 241-258.

[5] Kiyotaka Izumi , Keigo Watanabe, “Fuzzy behavior-based control trained by module learning to acquire the adaptive behaviors of mobile robots”, Mathematics and Computers in Simulation 51 (2000) 233–243.

[6] Lianfang Tian, Curtis Collins, “An effective robot trajectory planning method using a genetic algorithm", Mechatronics 14 (2004) 455–470.

[7] Malrey Lee, “Evolution of behaviors in autonomous robot using artificial neural network and genetic algorithm”, Information Sciences 155 (2003) 43–60.

[8] Kubota N., Morioka T., Kojima F., Fukuda T., “Learning of mobile robots using perception-based genetic algorithm”, [a]Fukui University ,3- 9- 1 Bunkyo, Fukui 910- 8507, Japan. [b]Osaka Institute of Technology, 5- 16- 1 Omiya, Asahi-ku, Osaka-city, Osaka, 535- 8585, Japan. [c]Kobe University, 1-1 Rokkodai-cho, Nada-ku, Kobe 657- 8501, Japan. [d]Nagoya University, 1Furo-cho, Chikusa-ku, Nagoya, 464- 8606, Japan.

[9] ROOMBA, website: www.gesolutions.net.

[10] Arredondo, T., Freund,W., Muñoz, C., Navarro, N., and Quirós, F.: Fuzzy Motivations for Evolutionary Behavior Learning by a Mobile Robot. In: Moonis, A., Dapoigny, R.(eds): Innovations in Applied Artificial Intelligence. Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin (2006) (Accepted paper in IEA/AIE06)