BULETINUL Universităţii Petrol – Gaze din Ploieşti	Vol. LXII No. 1/2010	119 - 126	Seria Tehnică

Neuro-fuzzy Controller for Mobile Robot

Cristina Popescu

Universitatea Petrol-Gaze din Ploieşti, Bd. Bucureşti, 39, Ploieşti
e-mail: cristinap@upg-ploiesti.ro

Abstract

A tendency for researchers is avoiding obstacles and reaching a target in navigation of mobile robots, but many results are obtained by simulation. In this paper a neuro-fuzzy based command for robot is proposed, first by designing a fuzzy controller for obstacle avoidance and after by using a Takagi-Sugeno controller, named neuro-fuzzy controller, which learn like a neural network. All the results are obtained by simulation by simulation in Matlab language and by experiments for implementation on the Khepera III mobile robot.

Key words: *mobile robot, neural network, fuzzy, neuro-fuzzy control.*

Introduction

Fuzzy logic control offers methods to control non-linear plants known to be difficult to model. In this category the mobile robot is included, so it is a good candidate as experimental platform for the validation of fuzzy and neuro-fuzzy controllers.

In this paper there were described some experiments with the Khepera III mobile robot developed in 2007 in the Microcomputing Laboratory at the Swiss Federal Insitute of Technology. It was implemented on this robot a fuzzy and a neuro-fuzzy controller, which are first validated by simulation in Matlab language. This problem was similarly studied for Khepera II mobile robot [1], but for Khepera III the sensorial system was changed, so the structure to control the robot was changed, too. For implement the simulation results it was developed a Matlab function to communicate with the robot, via bluetooth connection, using a Windows platform. By changing or adapting the sensorial system of the robot, one can implement more behavioral types for the robot, such as target reaching or a line following.

This paper is an extension of paper *Implementation of neuro-fuzzy controller on Khepera III mobile robot*, published in Bulletin of Petroleum-Gas University of Ploiesti in 2008 [3] and presents the final results obtained in the work with Khepera III.

Khepera III Mobile Robot

KHEPERA III mobile robot is cylindrical in shape, measuring 130 mm in diameter and 70 mm in height and its weight is 690 g.

Fig.1. Details of Khepera III mobile robot

The configuration of this mobile robot is composed of DsPIC 30F5011 at 60 MHz processor, system and user memory, extension busses and a serial link. The microcontroller includes all the features needed for interfacing with memories, I/O ports and external interruptions sources.

The sensorial/motor board includes two DC motors with incremental encoders, 9 infrared proximity and ambient light sensors with up to 25 cm in range, 2 infrared ground proximity sensors for line following applications and 5 ultrasonic sensors with a range of 20 cm to 4 meters.

Khepera mobile robot is an educational robot and it is ideal to implement artificial intelligent techniques that are used for robot control. This paper is focused on Khepera mobile robot control, aiming to implement the obstacles avoidance behavior by using a Takagi-Sugeno neuro-fuzzy controller.

Fuzzy Controller for Implementing Obstacle Avoiding Behavior

It was supposed that we have all necessary information needed for robot navigation. The robot has 9 infrared proximity sensors and 2 motors. The inputs are the linguistic variables: *distances between the robot and the obstacle* and the outputs are the linguistic variables: *motor speeds*. Four input linguistic variables were used: *distance to the left* D_s, *distance to the right* D_d, *distance to the front* D_f and *distance on the back* D_{sp} , as shown in figure 2.

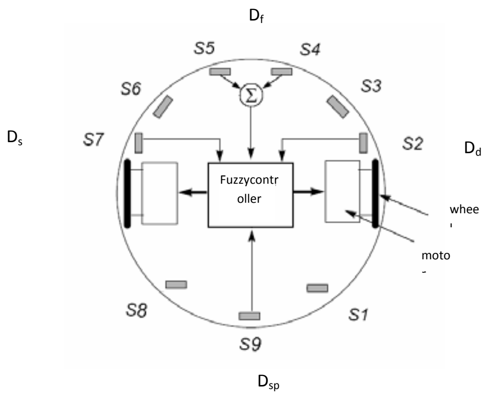


Fig.2. Inputs variables of Khepera III

We note with S_1, ..., S_{11} the sensor values normalized within [0,1], so the input variables are calculated in the following way:

$$D_s = S_7 \quad ; \tag{1}$$

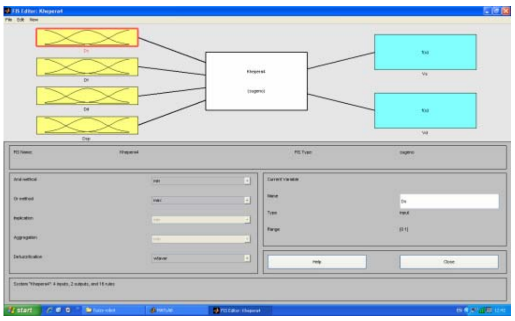
$$D_f = \frac{S_4 + S_5}{2} \quad ; \tag{2}$$

$$D_d = S_2 \quad ; \tag{3}$$

$$D_{sp} = S_9 \quad . \tag{4}$$

For each input linguistic variable, 3 linguistic values were defined: *small*, *medium*, *big*, while for each output we define 7 linguistic values: *backward fast*, *backward medium*, *backward slow*, *stop*, *forward slow*, *forward medium*, *forward fast* [2].

The aim of the first experiment is the implementation of a Sugeno fuzzy controller which allows navigation with obstacles avoiding behavior. For this controller, with 4 inputs and 2 outputs, the Matlab implementation offers a user interface shown in figure 3.



FIS Name: Khepera4 FIS Type: sugeno

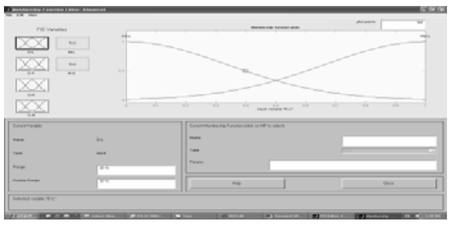
And method: min
Or method: max
Implication: min
Aggregation: max
Defuzzification: wtaver

Current Variable
Name: Ds
Type: input
Range: [0 1]

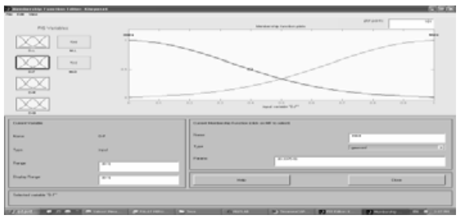
System "Khepera4": 4 inputs, 2 outputs, and 18 rules

Fig. 3. Fuzzy controller with 4 inputs and 2 outputs.

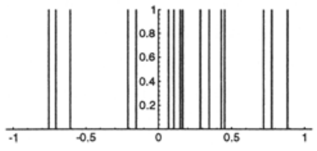
Each membership function for input variables is of Gaussian type while for output variables singleton functions were used, like in figure 4.



a) Distance to the left D_s



b) Distance to the front D_f



c) Motor speed v_s

Fig.4. Membership functions for input and output variables.

The initial rules base describing obstacle avoiding behavior contains 16 rules and is presented in table 1. It was demonstrated that with 16 rules, Khepera avoid obstacles with success, but adding rules can lead to an inconsistent rules base or, worse, conflicts between rules can appear.

Table 1. The initial base rules of the controller.

Rule	**D_s**	**D_f**	**D_d**	**D_{sp}**	**v_s**	**v_d**
R1	Small	Small	Small	Small	Stop	Stop
R2	Small	Small	Small	Big	Backward fast	Backward slow
R3	Small	Small	Big	Small	Backward slow	Forward fast
R4	Small	Small	Big	Big	Forward medium	Backward medium
R5	Small	Big	Small	Small	Backward fast	Backward fast
R6	Small	Big	Small	Big	Forward medium	Stop
R7	Small	Big	Big	Small	Backward slow	Stop
R8	Small	Big	Big	Big	Forward medium	Forward medium
R9	Big	Small	Small	Small	Backward fast	Forward fast
R10	Big	Small	Small	Big	Forward medium	Forward fast
R11	Big	Small	Big	Small	Forward slow	Backward medium
R12	Big	Small	Big	Big	Backward slow	Backward slow
R13	Big	Big	Small	Small	Backward slow	Forward medium
R14	Big	Big	Small	Big	Backward medium	Backward slow
R15	Big	Big	Big	Small	Stop	Backward slow
R16	Big	Big	Big	Big	Backward medium	Forward medium

For defuzzification it was used the weight average method and the outputs were calculated as follow [6]:

$$y_1 = \sum_{n=1}^{N} \overline{u}_n w_{n1} = \frac{1}{\sum_{n=1}^{N} u_j} \sum_{n=1}^{N} u_n w_{n1} \tag{5}$$

$$y_2 = \sum_{n=1}^{N} \overline{u}_n w_{n2} = \frac{1}{\sum_{n=1}^{N} u_j} \sum_{n=1}^{N} u_n w_{n2} \tag{6}$$

where y_1, y_2 are the outputs controller, u_n are the firing strengths for the n rule and w_{n1}, w_{n2} are the parameters of n rule for the two outputs.

With this rules base, the fuzzy controller was implemented with success on the real robot Khepera, the communication between the robot and the computer being made by Bluetooth technology. Figure 5 is suggestive for obstacle avoiding behavior.

Fig.5. Obstacle avoiding behaviour with fuzzy controller.

Neuro-fuzzy Controller for Implementing the Obstacle Avoiding Behavior

The characteristic of the Takagi and Sugeno's controller is that the consequent parts of linguistic rules are expressed as functions of linguistic variables, and in this case we will consider only the special type of Takagi and Sugeno's controller were these functions are constants [3].

The Takagi Sugeno controller's parameters will be adapted by a supervised learning method based on gradient descent, which consists of modifying parameters in order to obtain the desired output in response to given inputs. A supervised learning method is an identification algorithm. An off-line identification algorithm was chosen as supervisor the fuzzy controller implemented earlier was used. After learning, linguistic rules are extracted from the system parameters. They may be different than the initial rules [4].

The Takagi and Sugeno's fuzzy controller has three types of parameters to adapt:

- centre values

$$a = (a_{11}, \ldots, a_{nm}, \ldots, a_{NM})^T, \tag{7}$$

- width values

$$b = (b_{11}, \ldots, b_{nm}, \ldots, b_{NM})^T \quad , \tag{8}$$

- consequent values

$$c = (c_{11}, \ldots, c_{nk}, \ldots, c_{NK})^T \quad . \tag{9}$$

The recursive learning rules are given by [5]:

$$a_{nm}(t+1) = a_{nm}(t) - \Gamma \frac{\partial V(z)}{\partial a_{nm}} \tag{10}$$

$$b_{nm}(t+1) = b_{nm}(t) - \Gamma \frac{\partial V(z)}{\partial b_{nm}} \tag{11}$$

$$c_{nk}(t+1) = c_{nk}(t) - \Gamma \frac{\partial V(z)}{\partial c_{nk}} \tag{12}$$

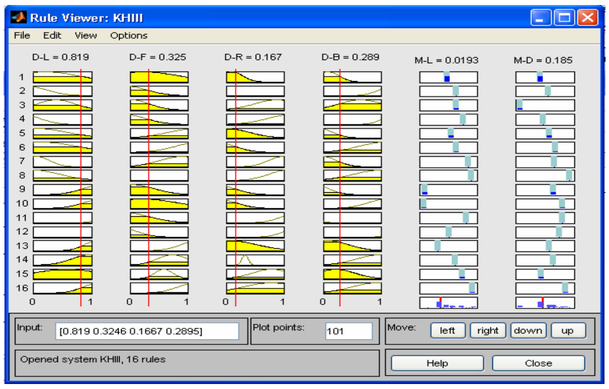
For Gaussian membership functions of the controller, the parameters and weights adapting is done by:

$$a_{nm}(t+1) = a_{nm}(t) - \Gamma_a \frac{u_n}{\sum_{n=1}^{N} u_n} \frac{x_m(t) - a_{nm}(t)}{b_{nm}(t)^2} \sum_{k=1}^{K} \left(y_k(t) - y_{d_k}(t) \right) \left(c_{nk}(t) - y_k(t) \right) \tag{13}$$

$$b_{nm}(t+1) = b_{nm}(t) - \Gamma_b \frac{u_n}{\sum_{n=1}^{N} u_n} \frac{(x_m(t) - a_{nm}(t))^2}{b_{nm}^3} \sum_{k=1}^{K} \left(y_k(t) - y_{d_k}(t)\right)\left(c_{nk}(t) - y_k(t)\right) \tag{14}$$

$$c_{nk}(t+1) = c_{nk}(t) - \Gamma_c \frac{u_n}{\sum_{n=1}^{N} u_n}\left(y_k(t) - y_{d_k}(t)\right) \tag{15}$$

In time of the learning process the rules base changes and one can observe there are no contradictions. It means that are not to be found two rules with the same antecedents parts and different consequent parts. In Matlab the modification of the rule base is shown in figure 6.



Input: [0.819 0.3246 0.1667 0.2895]

Plot points: 101

Opened system KHIII, 16 rules

Fig.6. Matlab viewer for modified rules base.

As it can be seen in figure 6, during the learning process, the rules base is changed, a subset rules (before and after learning process) being shown in table 2 and table 3.

Table 2. Rules before learning.

	dist. to the left D_s	dist. to the front D_f	dist. to the right D_d	dist. to the back D_{sp}	left speed V_s	right speed V_d
1	small	small	small	small	0.5	0.30
3	small	small	big	small	0.3	-0.3
11	big	small	big	small	0.90	0.77
16	big	big	big	big	0.40	0.63

Table 3. Rules after learning

	dist. to the left D_s	dist. to the front D_f	dist. to the right D_d	dist. to the back D_{sp}	left speed V_s	right speed V_d
1	smaller than medium	somehow small	small	small	-0.02	0.10
3	somehow medium	small	biger	somehow medium	0.33	-0.44
11	-	small	very big	-	0.70	0.69
16	very big	very big	biger than medium	big	0.91	0.75

The initial weights of neuro-fuzzy controller were initialized with stochastic values between 0 and 1. It was observed that the 16th rule is activated when the robots does not see any obstacle and a rapid movement onward were generated. The 11th rule was activated when the robot sees an obstacle in front and a movement straight ahead was generated. This rule cannot be individual interpreted because usually more rules at same time are activated.

The Min-Max method for the inference process and Centre of Gravity method for defuzzification were used. After learning process the membership functions are modified like in the following diagrams (figure 7):

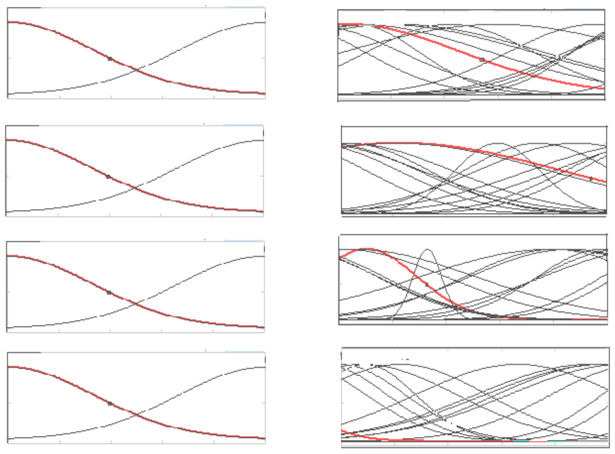


Fig. 7. Membership functions before and after learning.

Using simulation techniques (in Matlab language), it is observed that the robot learns the desired behavior with the 16 proposed rules, after 500 iterations as figure 8 depicts.

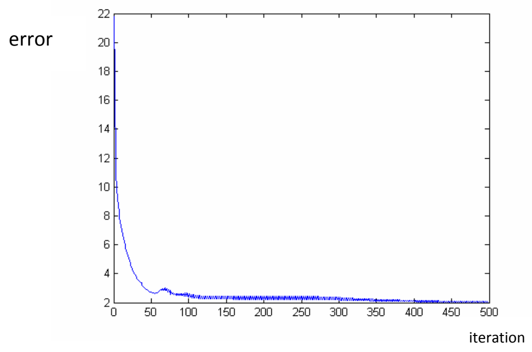


Fig.8. Evolution of the error.

The implemented neuro-fuzzy controller was tested on the Khepera III mobile robot. By using this practical implementation, it has been proved that the controller parameters adapting, as well as the linguistic rules modification were accomplished, the robot successfully avoiding the obstacles in the test scene.

By an adequate modification of the rules base on the mobile robot Khepera the wall or black line following behavior could be implemented, the robot being equipped with all required sensors.

It was demonstrated that the proposed method for analyze and design the neuro-fuzzy controller could be implement to a real robot.

Conclusion

This paper presented the design and implementation of Takagi and Sugeno's controller named neuro-fuzzy controller. For the first time, this problem was studied by Godjevac (1997) [1] [2], but the method and the algorithm was adapted for the new Khepera III mobile robot.

The first part presented the Khepera III configuration. Consequently, fuzzy controller with 16 linguistic rules which lead to obstacles avoiding behavior was implemented.

It was observed that some problems in robot navigation could appear when more than 20 linguistic rules are implemented, because there is the risk to create an inconsistent rules base and set some erroneous parameters for fuzzy controller.

The second part of this paper presents the implementation of a neuro-fuzzy controller which uses as professor the fuzzy controller earlier implemented. The proposed methodology allows introduction of some initial information about the robot inside the controller. During the learning step, the parameters are adjusted, the extracted rules from these parameters helping the designer to learn more about the robot and the controller.

This paper also offers a number of possible directions of future work. First, the use of supervised learning techniques can be studied. These techniques could avoid the learning from a supervisor and completely automate the design procedure. Second, more complicated tasks for mobile robot should be investigated.

References

1. Godjevac, J. – *Neuro-Fuzzy Controllers Design and Application*, Presses Polytechniques et Universitaires Romandes, Laussane, 1997;
2. Godjevac, J., Steele, N. – Neuro-Fuzzy Control for Basic Mobile Robot Behaviours, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*, Driankov, D., Saffiotti, A. (eds.), Physica-Verlag Heidelberg, 2001, pp. 97-117;
3. Popescu, C., Cangea, O. – Implementation of neuro-fuzzy controller on Khepera III mobile robot, *Buletinul Universității Petrol-Gaze din Ploiești - Seria Tehnică*, Ploieşti, 2008;
4. Popescu, C. – *Neuro-Fuzzy Control for Mobile Robot*, Ph.D. Thesis, U.P.G. Ploieşti, 2008;
5. Steele, N., Godjevac, J. – Adaptive Radial Basis Function Neural Networks and Fuzzy Systems, *Conference on Computational Engineering in Systems Applications*, CESA'96, Lille, France, pp.143-148;
6. Sugeno, M., Murakami, K. – An Experimental Study on Fuzzy Parking Control Using a Model Car, *Industrial Applications of Fuzzy Control*, Sugeno, M. (ed.), North-Holland, 1985, pp.125-138.

Conducerea neuro-fuzzy a robotului mobil

Rezumat

În prima parte a acestei lucrări a fost prezentată conducerea robotului mobil Khepera III în faza de ocolire a obstacolelor, utilizând un regulator fuzzy. Rezultatele au fost obținute întâi prin simulare în limbajul Matlab, fiind validate ulterior prin implementarea fizică pe robotul mobil. In partea a doua, s-a realizat conducerea neuro-fuzzy a robotului mobil pentru implementarea aceluiaşi comportament, în condițiile folosirii regulatorului fuzzy implementat anterior ca profesor. De asemenea, întâi a fost utilizată simularea în Matlab, după care a fost implementat cu succes regulatorul neuro-fuzzy şi pe platforma reală.