EVOLUTIONARY LEARNING OF A FUZZY CONTROLLER FOR A MOBILE ROBOT

FADI HALAL[1], I. DUMITRACHE[2]

Tehnicile inteligente bazate pe logica fuzzy, rețelele neurale şi algoritmi genetici sunt folosite cu mult succes în conducerea roboților autonomi. Sistemele hibride bazate pe combinații ale acestor tehnici pot maximiza eficiența acestor tehnici. În această lucrare, se prezintă un sistem hibrid geno-fuzzy care foloseşte un algoritm genetic pentru optimizarea unui sistem de conducere cu logică fuzzy pentru un robot Khepera care trebuie să atingă un anumit punct în spațiul de lucru pe traseul cel mai scurt. Algoritmul genetic optimizează funcțiile de apartenență şi generează reguli optimale. Rezultatele prezentate în această lucrare demonstrează validitatea abordării hibride bazată pe combinații ale tehnicilor inteligente de conducere.

Fuzzy control systems, neural networks and genetic algorithms can be cooperatively used for designing robot control systems. This paper presents a hybrid geno-fuzzy system based on a genetic algorithm that optimizes the membership functions and the rule structure of a fuzzy controller. The robot is a Khepera mobile robot that has to follow a track and find a target. The presented results demonstrate the validity of such a hybrid approach.

Keywords: geno-fuzzy system, fuzzy logic, genetic algorithm, mobile robot.

1. Introduction

Intelligent robots sharing city roads with humans and other vehicles is not a simple dream. Autonomous driving will enable a robot vehicle to drive independently along the road. [1] In this paper we used fuzzy control to drive a Khepera robot on a given in a simulation environment named Kiks. The first results have given us the reason to apply a hybrid geno-fuzzy control system, which has successfully driven the robot along the track. The geno-fuzzy system improves the design process and the performance of the fuzzy control system. [2] The first section presents the robot control architecture. The second section shows the fuzzy control system while the third section explains how to apply geno-fuzzy

[1] PhD student, Dept. of Automatic Control and System Engineering , Faculty of Automatic and Computer Science, University POLITEHNICA of Bucharest, Romania, fadi_halal@yahoo.com
[2] Prof, Dept. of Automatic Control and System Engineering , Faculty of Automatic and Computer Science, University POLITEHNICA of Bucharest, Romania

system. We developed the experiment and compared the results in the fourth section.

2. Robot control architecture

The control of a robot under parameter variations and load disturbances is an important problem [3]. Fig. 1 is illustrating an approach to build a control algorithm for a mobile robot, which is the "sense- plan-act" architecture. The robot problem was decomposed into a vertical series slice, which has the following functionalities:

- Observe the surrounding environment
- Make an internal plan of the area
- Adapt the robot plan
- Execute the plan
- Create a new plan when some thing was changed

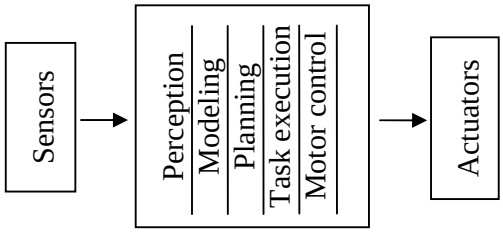


Fig.1. Traditional decomposition of a mobile robot control system into function modules.

In 1986 Rodney Brooks came with a new approach, which decompensate the problem into behaviors instead of function components [4], and this is illustrated into horizontal series slice, as shown in Fig. 2. Behaviors could be obstacle avoidance, wall-following, exploration or target seeking. A certain number of behaviors run as parallel processes, while each behavior can access all sensors, only one behavior can have control over the robot actuators.

In competitive control methods only one behavior affects the motor output of the robot in a particular moment. In cooperative control methods different behaviors may contribute to a single motor action although with different strength [5].

We decided to use a fuzzy control system to handle behavior selection, for controlling a Khepera mobile robot. This control structure has 3 sensor inputs which are S_{left}, S_{front} and S_{right}, corresponding to the sensors on the left, front and right hand side of the robot. This control will generate two outputs that are left motor speed and right motor speeds, respectively named LMS and RMS; these variables select the currently active behavior and cause a robot action. The control

system it self developed with genetic algorithm designed to optimize a fitness function describing the task criteria.

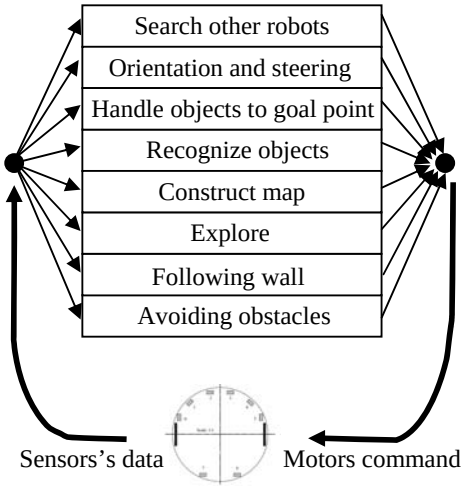


Fig.2. A decomposition of a mobile robot control system based on task achieving behaviors.

3. Fuzzy control system

3.1. Fuzzy controller design

The control input variables are the six sensors input (S0 …S5), and robot's coordinate. We ignored the two back sensors input (S6 and S7) that have no effect on the fuzzy control. The output variables are the left motor speed and right motor speed (LMS and RMS).

Sensors simplification was used as follows to reduce the number of the sensor inputs:

S_{left} = ((S0 +S1)/2)

S_{front} = ((S2+ S3)/2)

S_{right} = ((S4+S5)/2)

as shown in Fig. 3 [5]. Each input has three trapezoidal linguistics membership functions, which are called *near*, *med*, and *far*, as shown in Fig. 4.A, denoting the distance from an obstacle. These inputs have the same membership function shape and design for each sensor input.

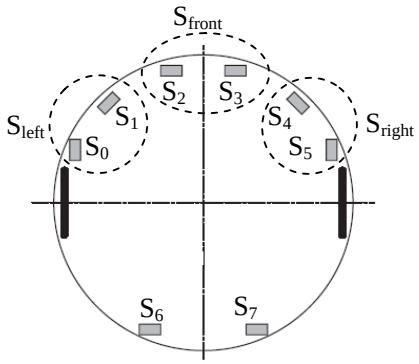
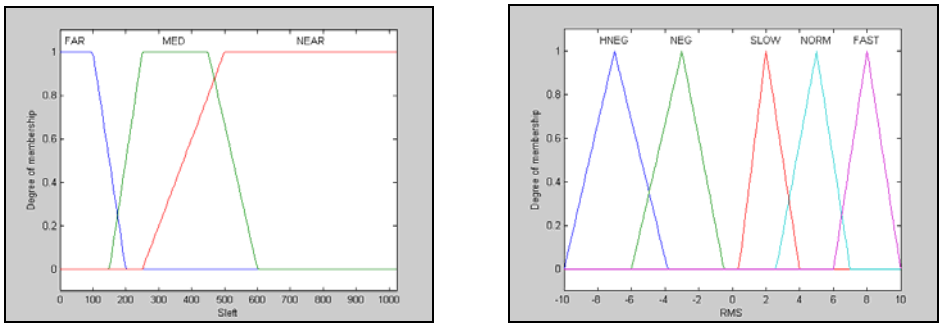


Fig.3. Simulation model of the Khepera robot

The output variables are LMS and RMS, respectively. Each output has five triangular membership functions, that are named as (*hneg*, *neg*, *slow*, *norm*, *fast*) representing the motor speed, as shown in Fig. 4.B. The fuzzy control system has 18 rules representing the robot behaviors, like left wall following, right wall following, walk through the corridor, obstacle avoidance, steering and tracking behavior.



(A) (B)

Fig.4. A: *Sleft* input membership function, B: *RMS* output membership function

3.2. Genetic representations of the fuzzy controller

We have encoded the rule base into a chromosome for a genetic algorithm in order to optimize these rules. We encoded the membership functions in each sensor input (Far, Med and Near), respectively coded as 1, 2 and 3, and for each output membership function (Hneg, Neg, Slow, Norm, and Fast), respectively coded as 1, 2, 3, 4 and 5. Fig. 5 illustrates the encoded input membership function, and the encoded output membership function.

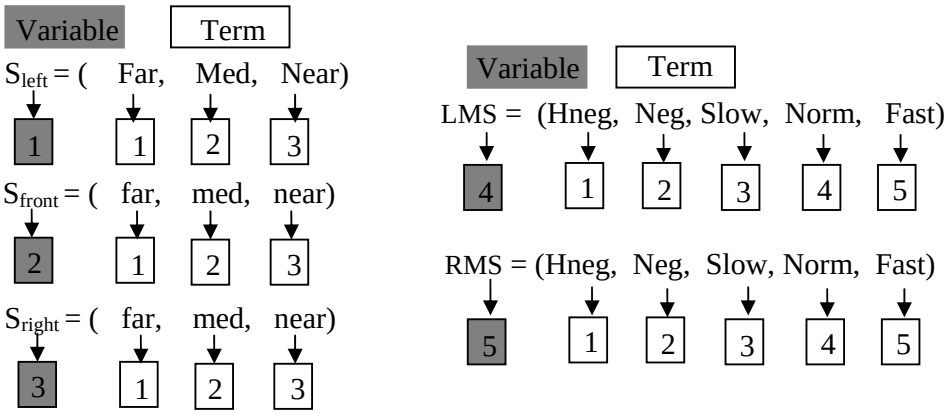


Fig.5. The input and the output membership function encoded

For example, the rule *If* S_{left} *= near and* S_{front} *= far and* S_{right} *= med Then LMS = fast and RMS = norm*, can be encoded as a string vector 3 1 2 5 4 and the chromosome is illustrated as a string vector, as shown in Fig. 6.

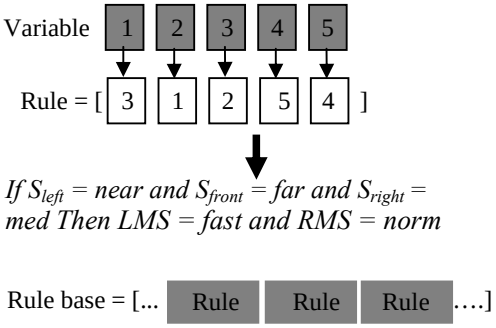


Fig.6. The rules base chromosome encoded

3.3. Rule base

The rule set had to be simplified. This simplification was accomplished by eliminating rules with low or even no probability to occur, and rules that cause the same effect in the robot movement. The final rules base is presented in table 2 which has been divided into eight basic groups: straight movement, when the robot has either no obstacle in the target direction or the obstacle is far; walk through the corridor, where the robot walks along the corridor; left wall following, where the robot followed the left wall; right wall following, where the robot

followed the right wall, while avoiding obstacles behaviors were divided into four groups: Avoiding left front obstacles, Avoiding right front obstacles, Avoiding front obstacle, Avoiding blocked zone.

Table 1

Fuzzy rule base encoded

S_{left}	S_{front}	S_{right}	LMS	RMS	Robot's behaviors
1	1	1	5	5	straight movement
2	1	2	4	4	walk along the corridor
3	1	3	3	3	
2	1	3	3	4	
3	1	2	4	3	
2	1	1	4	2	left wall following
3	1	1	5	1	
1	1	2	2	4	right wall following
1	1	3	1	5	
2	2	1	4	2	avoiding left front obstacles
3	3	1	5	1	
3	2	1	5	2	
1	2	2	2	4	avoiding right front obstacles
1	3	3	1	5	
1	2	3	2	5	
1	2	1	1	5	avoiding front obstacle
2	2	2	2	2	avoiding blocked zone
3	3	3	2	2	

4. Experiments

In this paper we used a genetic algorithm to optimize the performance of the fuzzy system. Table 2 shows the advantages and the disadvantages for fuzzy systems and genetic algorithms. The genetic algorithm was used for its ability to learn [2]. Fig. 8 shows the structure of the hybrid geno-fuzzy control system that was used to control the robot.

Table 2

Advantages and drawbacks of fuzzy logic and GAS

Properties	Fuzzy systems	Genetic algorithm
Store knowledge	Explicit	None
learns	No	Ability to learn
Optimizes	None	Powerful
Fast	Yes	Yes
Handle nonlinearity	Yes	Yes

The general system architecture is composed of the mobile robot, the fuzzy control system, the evolution strategy that adapts the fuzzy membership functions and the rules base, a simulation environment (KIKS simulator for Khepera robots), a fitness function to evaluate the quality of robot behaviors, as is shown in Fig. 7. The environment is shown in Fig. 8, where the control task is to drive the robot along the grey track in order to reach the target point as fast as possible.

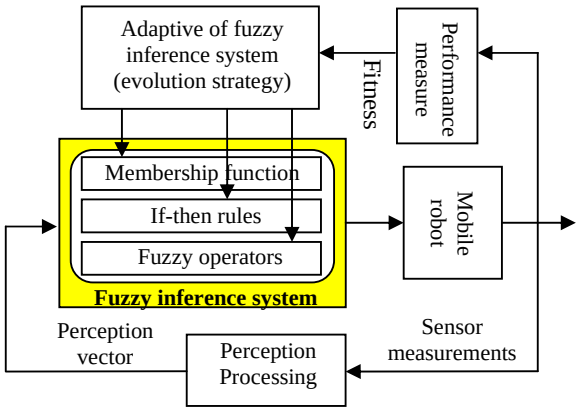


Fig.7. Architecture of the geno-fuzzy system to control a mobile robot.

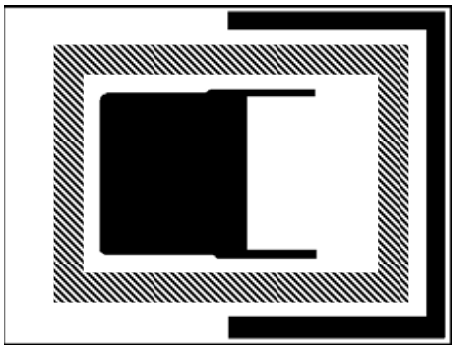
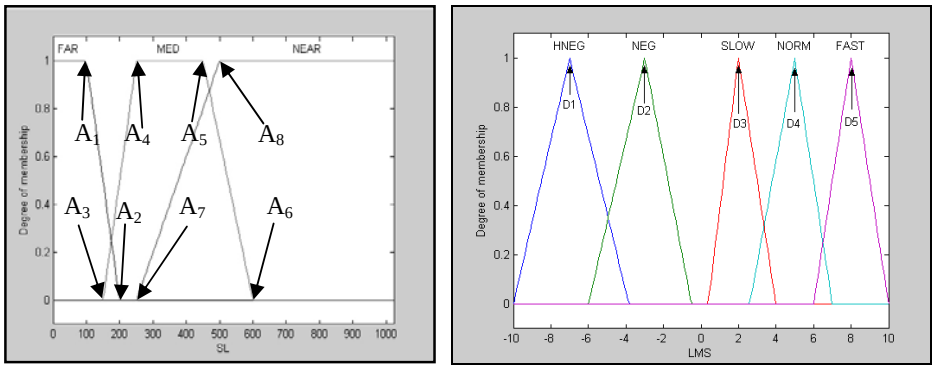


Fig.8. Simulation environment

4.1. Chromosome

In this paper is presented a novel chromosome encoding algorithm used to optimize the membership functions and the output rule base. So an optimal fuzzy control system is obtained which drove the Khepera mobile robot to achieve its target with good performance and optimal behaviors. We have encoded 34 parameters from the input and output membership function to form a chromosome

segment in order to optimize these functions shape. Fig. 9 shows the eight genes that were encoded from S_{left} input function; these genes are called respectively A_1, A_2, A_3, A_4, A_5, A_6, A_7 and A_8. From the S_{front} and S_{right} input we encoded the two other chromosomes segment B and segment C, where each of them contain eight genes. They are named respectively B_1, B_2, B_3, B_4, B_5, B_6, B_7, B_8, C_1, C_2, C_3, C_4, C_5, C_6, C_7 C_8, and Fig. 10 shows the membership functions chromosome segment. The genes D_1, D_2, D_3, D_4, D_5, E_1, E_2, E_3, E_4, and E_5 are encoded for tuning respectively the LMS and RMS output membership functions.



A_1	….	A_6	A_7	A_8	B_1	B_2	..	B_8	C_1	C_2	…	C_8	D_1	...	D_5	E_1	…	E_5

Fig.9. Chromosome's segment for encoding membership functions

In our fuzzy system we suppose 18 antecedence rules and the genetic algorithm optimizes the output of the rules base. The inputs values are predictable because the robot moved in its simulation environment. The chromosome output rules segment contained 36 genes that present the supposed LMS linguistic term and RMS linguistic term, as shown in Fig. 10.

F_1	F_2	F_3	F_4	…	…	F_{33}	F_{34}	F_{35}	F_{36}
Output rule 1		Output rule 2				Output rule 17		Output rule 18	

Fig.10. Chromosome Segment F

Fig.11 shows the whole chromosome and his genes.

A_1	--	A_8	B_1	--	B_8	C_1	--	C_8	D_1	--	D_5	E_1	--	E_5	F_1	F_2	--	F_{35}	F_{36}

Fig.11.The whole chromosome

4.2. Multipoint crossover

The multi-point crossover is the best genetic operator method that can be used in this problem in order to increase the number of string segments exchanged. The parent chromosomes, P_1 and P_2, are cut virtually at multiple random locations, and the portions of the chromosome between the cuts were exchanged. The result is two offspring I_1 and I_2, as is shown in Fig. 12. We used multi-point crossover because the genes have integer values, the genes values of the output rules base are between 1 and 5, whereas the genes values of membership function chromosome segment are between 100 and 800, depending on the membership function itself. On the other hand, the genes had bounds in case to keep overlaps between the membership functions and the search for output rule base will be heuristic.

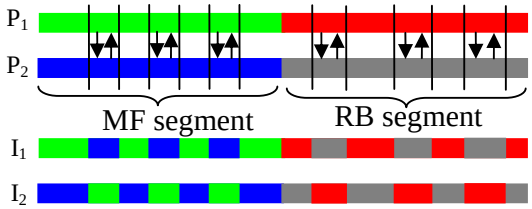


Fig.12. Multi-point crossover

4.3. Fitness function

This fitness function is a performance criterion that evaluates the performance of each chromosome. Higher fitness values are better when we want to maximize the function [5]. In this paper the fitness function trains the fuzzy control to optimize the robot path, thus the robot moved along its track with performance behaviors, and with the suitable speed without any collision. Practical the evolution process optimized the membership functions shape and the output rule base of the fuzzy controller.

The fitness function is:

$$F = S + Bon + A + TR + TI$$

And is calculated and summed over 1300 robot steps. Of its five components, *S* improves the speed; *A* improves collision avoidance, *Bon* gives the robot a bonus when it walks along the desired track. These three components are calculated and summed each robot step. *TR* and *TI* teach the robot to get to its target in a the shortest time possible. The time is either the number of steps that the robot needs to get to its target or is 1300 steps. The values of the *TI* and *TR* will be 0 when the robot doesn't get its target.

<table>
<tr><td>$$S = \sum_{i=1}^{t} \frac{|M_L + M_R|}{2M_{MAX}} / t$$</td><td rowspan="6">Where:
- Mmax: maximum robot's speed (equal to 10);
- (M_L, M_R): left motor and right motor speed
- Smax: maximum sensor's reading (equal to 1023).
- S_t: proximity-sensor (S_{left}, S_{front}, S_{right}) highest activity at step t.
-t: the number of total steps</td></tr>
<tr><td>$$\text{Bon} = \sum_{i=0}^{t} \frac{pp}{1300}$$
pp= 1 if the robot is on it track
pp=0 1 if the robot isn't on it track</td></tr>
<tr><td>$$\text{A} = \left[1 - \sum_{i=0}^{t} \frac{S_t}{S_{MAX}} / t \right]$$</td></tr>
<tr><td>TR =1: If the robot gets the target
TR=0: If the robot doesn't get the target</td></tr>
<tr><td>$$TI = \left[\frac{1300 - t}{1300} \right]$$ If the robot gets the target
TI = 0: If the robot doesn't get the target</td></tr>
</table>

5. Experiments development and comparison of results

5.1 Experiments development

The genetic algorithm generates fuzzy parameters set for each population in any generation. The fuzzy controller drives the robot to make its task within a fixed time. The robot gets sensor data and then decides the suitable behavior. Avoiding obstacle will apply if there is an obstacle near the robot and if there is a wall then the robot will follow it. If the area is clear the robot seeks its target. The first priority is to keep the robot away from any obstacle, and then following wall

in navigation mode and seeking target behavior. A flow chart for geno-fuzzy control system design is given in Fig. 13.

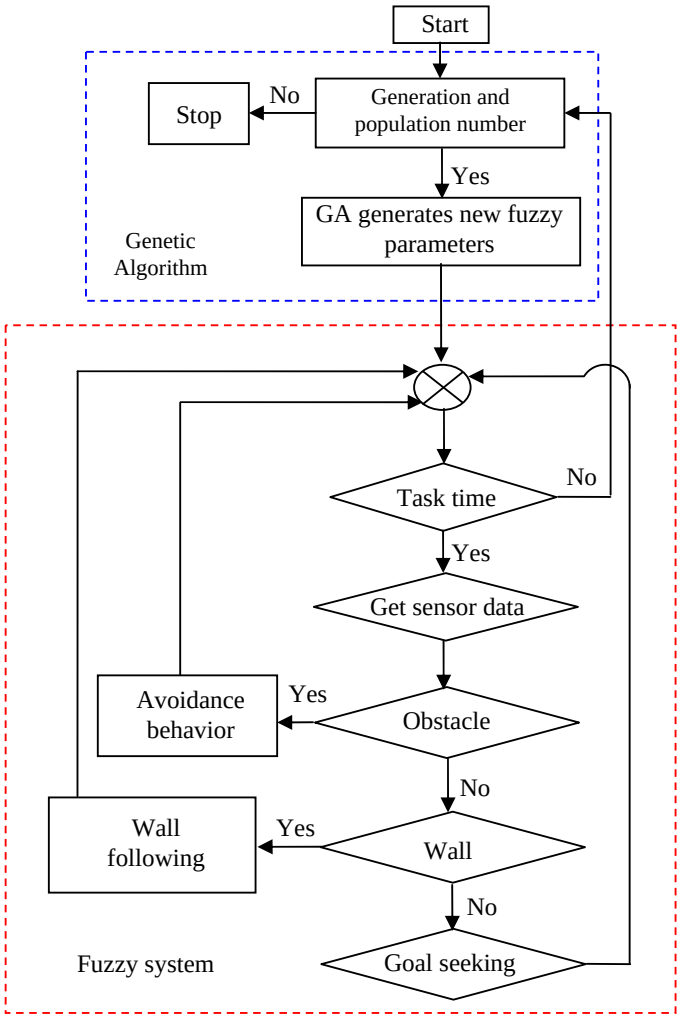
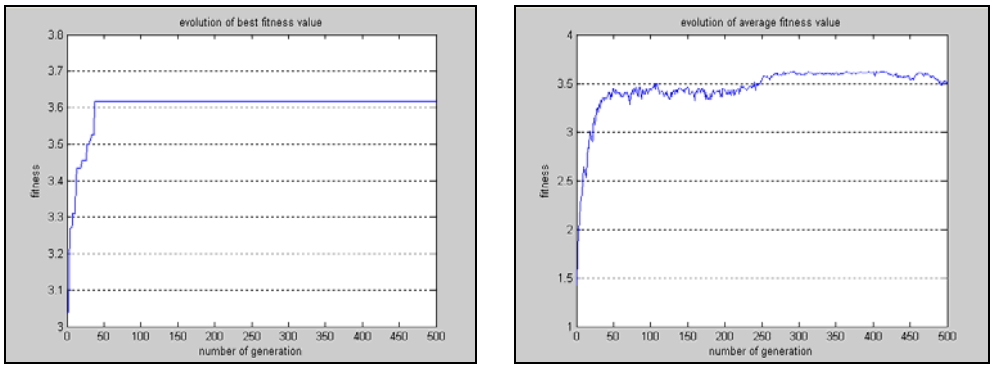


Fig.13. Flow chart for geno-fuzzy control system

We used the GAOT toolbox for Matlab. GAOT is a Genetic Algorithm Optimization Toolbox (GAOT) used for optimizing the fuzzy system. [20] In the evolution process we used the following parameters: Population size 50 individuals; crossover rate 80%; mutation rate 5%; number of generations 500.
Fig 14.A and 15.B show respectively the best chromosome fitness in each generation and the average of all the chromosomes in each generation.



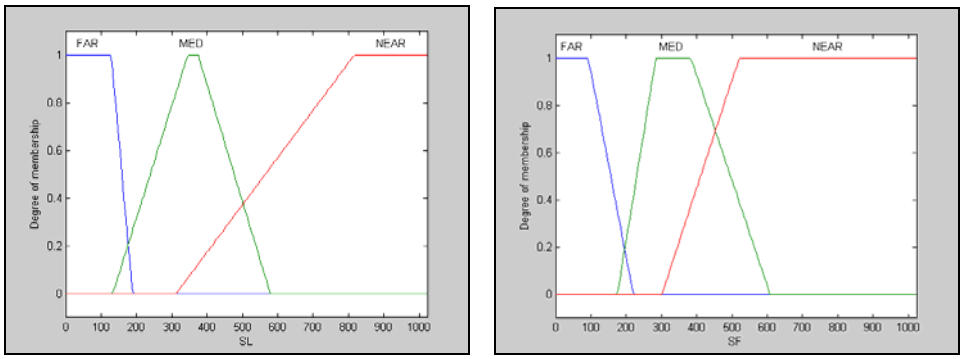
(A) (B)

Fig.14. A: Evolution of best chromosome, B: Evolution of average fitness chromosome

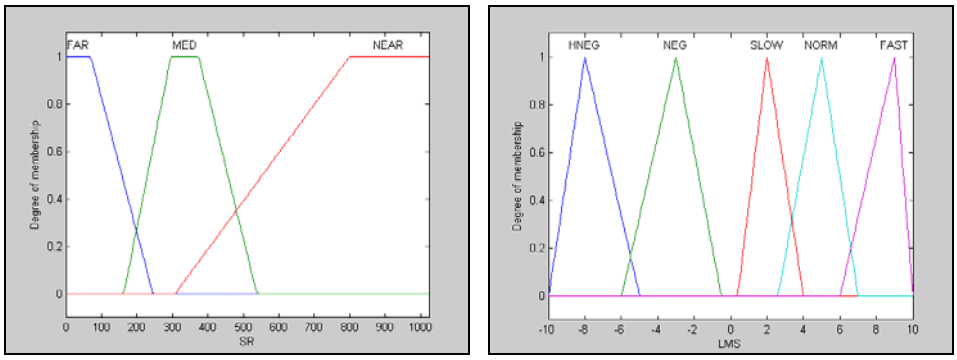
Figs. 15 (A, B, C, D, and E) show the optimal membership functions resulted after the evolutionary process. The figures are respectively for S_{left}, S_{front} and S_{right} sensor inputs, and for the LMS and RMS output membership functions. The optimal chromosome for the rule base output is as follows:

4 5	4 4	4 4	3 5	5 4	5 4	3 5	3 2	5 2
R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8	R_9
2 5	2 4	1 4	5 1	1 4	1 4	1 4	4 5	5 5
R_{10}	R_{11}	R_{12}	R_{13}	R_{14}	R_{15}	R_{16}	R_{17}	R_{18}

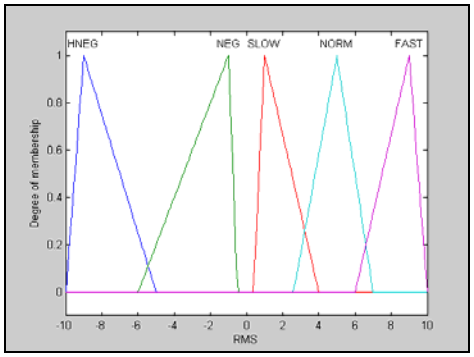
Where: 5, 4, 3, 2 and 1 represent respectively (HNEG, NEG, SLOW, NORM and FAST) that are the linguistic terms of the output fuzzy system.



(A) (B)



(C) (D)

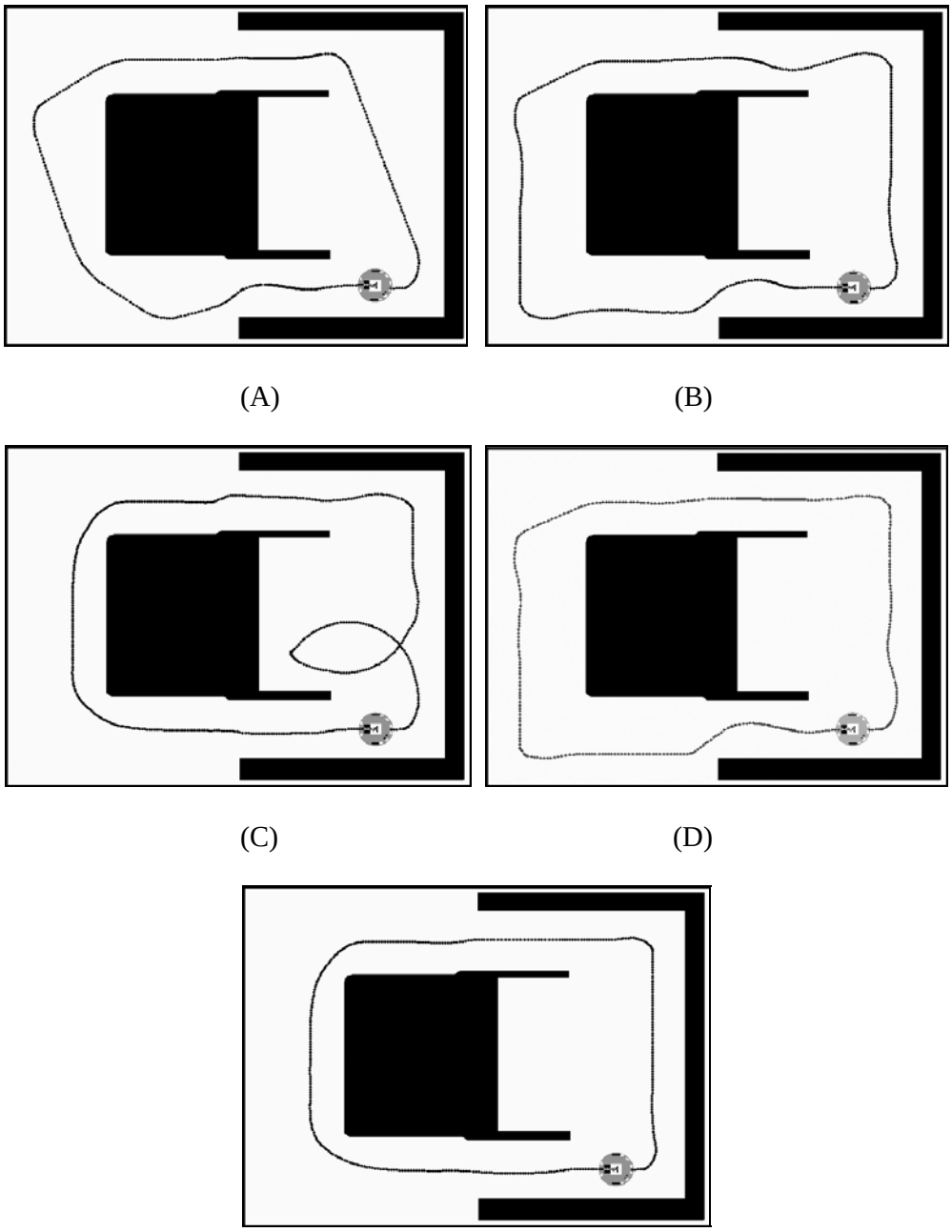


(E)

Fig.15. A: The optimal S_{left} input membership function, B: The optimal S_{front} input membership function, C: The optimal S_{right} input membership function, D: The optimal RMS output membership function, E: The optimal LMS output membership function

5.2 Comparison of results

We designed a fuzzy system to control the robot along a given track. The result was a poor performance of the robot, as are shown in Fig. 16(A and B), as the robot needs 105 seconds to get to the target, as shown in Fig. 16(C) and exceeded the track limits. The fuzzy system was modified and was tested in the Kiks simulation environment. Fig. 16(D) shows the robot's trajectory which improved as the robot needs now 55 seconds to get to the target. The results of the geno-fuzzy control systems are the best as presented in Fig. 16(E). The robot moved smoothly along its track and it needs 39 second to get to the target. Thus the geno-fuzzy system improves the path following behavior with a very short time to reach the target.



(A) (B)

(C) (D)

(E)

Fig.16. A and B: poor performance of the robot C: Robot with fuzzy system needs 105 seconds. D: Robot with modified fuzzy system needs 55 seconds, F: Optimal solution needs 39 seconds

6. Conclusions

In this paper we presented a hybrid geno-fuzzy control system for a mobile robot. Genetic algorithms are proven to be an efficient tool for designing an optimal fuzzy control system. The hybrid system optimized the membership function set and the output rule base on the fuzzy controller. The optimization improves the robot time performance and the robot behavior. The optimal solution has driven the robot on its track with a suitable speed. After the evolution process the robot walked fast along the corridor, its wall following ability improved significantly, while it managed to keep a suitable distance from the obstacles. Thus the optimal fuzzy system generated an optimal path towards the target.

R E F E R E N C E S

[1] *Jonathan Baber, Julian Kolodko, Tony Noel Michael Parent, and Liubo Vlacic*, cooperative autonomous driving intelligent vehicle sharing city roads, IEEE robotics & automation magazine 1070-9332/05/$20.00 ©2005

[2] *Fakhreddine O. Karray and Clarence De Silva,* Soft Computing and Intelligent Systems Design: Theory, Tools and Applications, Addison Wesley 2004 ISBN: 0321116178

[3] *Abduladheem A. Ali, Easa A. Abd,* Genetic-fuzzy inverse controller for a robot arm suitable for on line applications, transactions on engineering, computing and technology V9 November 2005

[4] *R. A. Brooks*, A robust layered control system for a mobile robot, IEEE Journal of Robotics

[5] *Stefano Nolfi, Dario Floreano*, evolutionary robotics: the biology intelligence and technology of self organizing machines (intelligent robotics and autonomous agents), 2000 ISBN: 0-262-14070-5

[6] *A. Abraham,* Hybrid Intelligent Systems: Evolving Intelligence in Hierarchical Layers, Stud-Fuzz 173, 159–179 (2005) © Springer-Verlag Berlin Heidelberg 2005

[7] *Frank Hoffmann,* Evolutionary algorithms for fuzzy control system design, Proceedings of the IEEE, 2001, to appear in special issue on Industrial Innovation using Soft Computing.

[8] *Frank Hoffmann*, Soft Computing Techniques for the Design of Mobile Robot Behaviours, © Elsevier Science Inc., 1994, NY 10010 0020-0255/94/$6.00

[9] Michel Pasquier, Li How Lim, "fuzzy world 2-reactive beheaviours and environment exploration strategies" The fifth international conference on control, automation, robotics and vision (ICARCV 98)

[10] *Michael A. Lee and Hideyuki Takagi,* Integrating design stages of fuzzy systems using genetic algorithms This paper appeared in the Proc. of the 2nd Int'l Conf. on Fuzzy Systems (FUZZ-IEEE'93), Vol.1, pp. 612-617 (1993)

[11] *Randy L. Haupt, Sue Ellen Haupt*, Practical Genetic Algorithms, John Wiley & sons, inc., publication second edition 2004 ISBN: 0471455652

[12] *Simon X. Yang, Xiaochuan Wang, Guoyin Wang Max Q.-H. Meng* ,An Embedded Genetic Fuzzy Motion Controller for a Mobile Robot, Copyright @ 2005 IFAC

[13] *Yuchao Zhou*, an Area Exploration Strategy Evolved by Genetic Algorithm, Master Thesis, of Science Athens, Georgia 2005.

[14] *Franz Rothlauf,* Representations for Genetic and Evolutionary Algorithms, © Springer-Verlag Berlin Heidelberg 2 edition (March 3, 2006) ISBN: 354025059X

[15] *E. Tunstel, T. Lippincott, and M. Jamshidi*, Behavior hierarchy for autonomous mobile robots: Fuzzy-behavior modulation and evolution, Intl. Journal of Intelligent Automation & Soft Computing, vol. 3, no. 1, pp. 37–50, 1997

[16] *Hartmut surmann, michail maniadakis*, learning feed-forward and recurrent fuzzy systems: a genetic approach, journal of system architecture 47 (2001) 649-662

[17] *Alan C. Schultz and John J. Grefenstette,* using a genetic algorithm to learn behaviors for autonomous vehicles, In Proceedings of the AIAA Guidance, Navigation and Control Conference, Hilton Head, SC, August 10-12, 1992.

[18] *Frank Hoffmann, Gerd Pfister,* A New Learning Method for the Design of Hierarchical Fuzzy Controllers Using Messy Genetic Algorithms, 1995.

http://citeseer.ist.psu.edu/cache/papers/cs/1463/http:zSzzSzHTTP.CS.Berkeley.EDUzSz~fhoffmanzSzifsa95.pdf/hoffmann95new.pdf

[19] Khepera "User manual" Version 5.02 K-Team Lausanne, 12 March 1999

[20] http://www.ie.ncsu.edu/mirage/GAToolBox/gaot

[21] http://www.tstorm.se/projects/kiks/