# An Empirical Study for Fitness Function Selection in Fuzzy Logic Controllers for Mobile Robot Navigation

Lefteris Doitsidis & Nikos C. Tsourveloudis
Intelligent Systems & Robotics Laboratory
Department of Production Engineering & Management
Technical University of Crete
University Campus, Chania GR – 73100
GREECE
*{ldoitsidis,nikost}@dpem.tuc.gr*

***Abstract* – Fuzzy logic is widely used for mobile robot navigation. The main draw back of this approach is the *ad hoc* design of the controllers used. A popular method for the optimization of fuzzy logic controllers for the navigation of mobile robots is the use of genetic algorithms. An issue, in this procedure is the selection of the fitness function for the improvement of the behavior of a pre-designed controller. We analyze the factors that influence the evolution of the fuzzy controller based on the fitness function used and present some preliminary results. In order to validate our approach a test bed has been developed based in a low cost robot.**

## I. INTRODUCTION

Fuzzy logic techniques are commonly used for navigation of different types of robot vehicles [1]. The popularity of the use of fuzzy logic is based on the fact that it can cope with the uncertainty of the sensors and the environment really well. By using it, the robot vehicles are able to move in known or unknown environments, using control laws that derive from a fuzzy rule base. This base is consisted from a set of predefined IF-THEN rules, which remain constant as far as it concerns their structure during the operation of the robot. These rules as long as the membership functions of the input and output variables are usually designed *ad hoc* by human experts.

Several researchers have used fuzzy logic for the navigation of a mobile robot. In [2] a layer goal oriented motion planning strategy using fuzzy logic controllers has been proposed, which uses sub-goals in order to move in a specific target point. Another approach is presented in [3], where the authors propose a control system consisted of fuzzy behaviors for the control of an indoor mobile robot. All the behaviors are implemented as Mamdani fuzzy controllers except one which is implemented as adaptive neuro-fuzzy. In [4] a combined approach of fuzzy and electrostatic potential fields is presented that assures navigation and obstacle avoidance.

The main draw back of the approaches presented is that the design of the fuzzy logic controller is relied mainly on the experience of the designer. In order to overcome this problem several researchers have proposed tuning of the fuzzy logic controller based on learning methods [5] and evolutionary algorithms [6] - [11], in an attempt to improve the performance and the behavior of the robot.

In [6] a fuzzy logic controller for a khepera robot in a simulated environment was evolved using a genetic algorithm and the behaviors of the evolved controller were analyzed with a state transition diagram. The robot by using the evolved controller produces emergent behaviors by the interaction of the fuzzy rules that were produced from the evolution process. In [7] the authors proposed a three step evolution process to self-organize a fuzzy logic controller. The procedure initially tunes the output term set and rule base, then the input membership functions and in the third phase it tunes the output membership functions. Hargas *et al.* in [8] proposed a fuzzy-genetic technique for the on-line learning and adaptation of an intelligent robotic vehicle. In [9] the authors present a methodology for the tunning of the knowledge base of the fuzzy logic controller based on a compact scheme for the genetic representation of the fuzzy rule base. In [10] the authors present a scheme for the evolution of the rule base of a fuzzy logic controller. The evolution takes place in simulated robots and the evolved controllers are tested in a khepera mobile robot. Nanayakkara *et al.* in [11] present an evolutionary learning methodology of a fuzzy behavior based controller using a multi objective fitness function that incorporates several linguistic features. The methodology is compared with the results that derive by the use of a conventional evolutionary algorithm.

A main issue which is not addressed in the literature, is related to the selection of factors of the fitness function which is going to be used in the evolution of a fuzzy logic controller. The majority of the fitness functions used is hand formulated and usually task specified. This results to controllers which depend heavily on the overall design of the functions. An attempt to formulate a way of picking the suitable function for a task was made by Nolfi and Floreano in [12]. They proposed the concept of "fitness space", which provides a framework for describing and designing fitness functions for autonomous systems.

In this paper we will attempt to analyze the differences in the behavior of a real robot which are produced by the evolution of a fuzzy controller using different types of fitness functions. These are categorized into three broad categories and formally analyzed in section 2. We will use a genetic algorithm with exactly the same parameters in order to evolve the membership functions of a predefined fuzzy controller. An analysis of the results will be presented and the influence of different parameters in the fitness function

1-4244-0136-4/06/$20.00 ©2006 IEEE

will be identified.

The rest of the paper is organized as follows. In section 2 the main characteristics of the three different types of fitness functions are identified and there is an analytical description of the actual functions used. In section 3 the fuzzy logic controller which was evolved is presented together with the genetic algorithm used for evolution. In section 4 a custom made robotic vehicle is presented and in section 5 the experimental results are presented and analyzed. Finally in section 6, issues for discussion and further research are presented.

## II. FITNESS FUNCTION CATEGORIES

The choice of the fitness function is a fundamental issue for the evolution of the controller of a mobile robot. The overall behavior of the robot can be affected by the form that these functions might have. There are several types of fitness function proposed in the literature and each one of them is considering different factors.

These functions can be divided based on whether there is prior knowledge of the task they want to achieve and according to the behavior of the controller that they produce. We would consider three types of functions which are *aggregate*, *behavioral* and *tailored fitness functions.*

The term aggregate fitness describes functions that select only for high-level success or failure to complete a task without regard how the task was completed. The main advantage of this type of function is that reduces the injection of human bias in the evolution process since it's aggregating the evaluation of the robot's performance in a single success/ failure term. For the evolution of complex behaviors aggregate functions are less depended on the designer than the behavioral and even more the tailored as they are incorporating the knowledge of the designer and they guide the evolution process in a desired behavior.

The term behavioral fitness describes functions which have task-specified hand-formulated functions that measure various aspects of a robot's functionality. The distinctive feature of this class of functions is that they are made only of terms or components that select for behavioral features of a presupposed solution to a given task.

Finally the tailored fitness functions are the ones that are responsible of the evolution behaviors that are pre-recognized from the designer of the function. This type of functions combines elements from both the behavioral and the aggregate categories. Usually in tailored functions the aggregate terms are measuring a degree partial task completion in a way that inserts some degree of *a priori* knowledge.

In order to investigate how different types of fitness functions are influence the evolution of a fuzzy logic controller for a mobile robot three different fitness functions were used.

The first fitness function used was pure aggregate function which was measuring only how close the robot has went in a target position comparing to its initial position. During the evolution process the level of activation of the sensors or the number of collisions with the obstacle wasn't considered. The form of the function for the individual *i* of the generation *j* was:

$$f_i = \left( 3e^{2\left(\frac{d_{final}-d_{initial}}{d_{initial}}\right)} \right) \Big/ 7.3891, \quad (1)$$

where, $d_{final}$ is the final distance of the robot from a predefined target point and $d_{initial}$ the initial distance of the robot from a predefined target point.

The second fitness function used was a behavioral fitness similar to the one used in [15] for the evolution of a discrete time recurrent neural network. The fitness was considering the percentage of straight motion of the vehicle, the activation level of the sensors and the velocity in each wheel of the vehicle. This function wasn't considering the distance of the robot's position relative to the target position. The form of the function for the individual *i* of the generation *j* was:

$$f_i = mean\left(v_{_L}, v_R\right)\left(a_F + a_L + a_R\right), \quad (2)$$

where, $v_L$ the velocity of the left wheel of the robot , $v_R$ of the right and $a_F$, $a_R$, $a_L$ are the activation levels of the front, right and left sensors respectively. By the term activation level we consider the percentage that is calculated by the division of the actual sensor reading divided by its maximum value. The activation level of each sensor derives from the following equation:

$$a_{sensor} = \frac{sensor\_reading(t)}{\max sensor\ range}, \quad (3)$$

where, *sensor_reading( t)* is the reading at the time step $t$ of the experiment. If the activation level is 1 then the sensor is not having any obstacle in its field of view.

The third function used, was a tailored fitness function which measured how close the robot has went in a target position comparing to its initial position and the activation level of each sensor that the robot had. The form of the function for the individual *i* of the generation *j* was:

$$f_i = \left( \left( 3e^{2\frac{d_{final}-d_{initial}}{d_{initial}}} \right) \Big/ 7.3891 \right) + \left(a_F + a_L + a_R\right). \quad (4)$$

## III. DESIGN OF THE FUZZY LOGIC CONTROLLER AND OF THE EVOLUTION PROCESS

### A. Fuzzy Logic Controller

Based on previous work [14] we have designed a fuzzy logic controller with four inputs and two outputs. The inputs are the *heading error*, the *distance from obstacles* as it was measured from the front sonar sensor and *the distance from*

*obstacles* as it was measured from the left and the right infrared sensors. The outputs from the controller were the values for the movement of the left and right servos.

The inputs from the sensors are used in order to calculate the collision possibilities in three directions, which are front, left and right collision possibility. The heading error is calculated from the robot's current heading and the desired heading.

Implementation wise the input variable heading error is including four trapezoidal membership functions and one triangular membership function. The input variable front collision possibility includes three trapezoidal membership functions and the input variables left and right collision possibility are including two trapezoidal membership functions each.

The value of each distance input variable $d_i$ $i=1,...,3$ (corresponding to front, left , right area) are expressed by the fuzzy sets $C_i$, $A_i$ (corresponding to close and away) for the left and right area and $C_i$, $MD_i$, $A_i$ (corresponding to close, medium distance and away) for the front area. The values of the input variable *heading error, (he),* are expressed by the fuzzy sets *FL, L, AH, R, FR* (corresponding to far left, left, ahead, right and far right). The outputs of the fuzzy logic controller are the speeds of the left and the right servo motor. The membership functions describing the fuzzy sets for each output variable are *Z, S, M, H* (corresponding to zero, small, medium and high speed).

Each fuzzy rule $j$ is expressed as:

IF *he* is $HE_j$ AND $d_1$ is $D_{j1}$ AND $d_2$ is $D_{j2}$ AND $d_3$ is $D_{j3}$ THEN *lm* is $LM_j$ AND *rm* is $RM_j$

wrere $j=1,...,$ *number_of_rules*, $D_{ji}$ is the fuzzy set for $d_i$ in the $j^{th}$ rule which takes the linguistic value of $C_i$, $MD_i$, $A_i$ for $i=1$ (front area) and $C_i$, $A_i$ for $i=2,3$ corresponding to the left and right areas. $HE_j$ takes the linguistic values *FL, L, AH, R, FR* and finally $LM_j$ and $RM_j$ are taking the linguistic values *Z, S, M, H*.

The generic mathematical expression of the $j^{th}$ navigation rule is given by:

$$\mu_{R^{(j)}}(he,d_i,lm,rm)=\min[\mu_{HE^{(j)}}(he),\mu_{D_i^j}(d_i),\mu_{LM^{(j)}}(lm),\mu_{RM^{(j)}}(rm)]. \quad (5)$$

The overall navigation output is given by the max-min composition and in particular:

$$\mu_N^*(lm,rm)=\max_{he,d_i}\min[\mu_{AND}^*(he,d_i),\mu_R(he,d_i,lm,rm)], \quad (6)$$

where , $\mu_R(he,d_i,lm,rm)=\bigcup_{j=1}^{J}\mu_{R^{(i)}}(he,d_i,lm,rm)$ .

*B. Genetic Fuzzy Tunning*

For the evolution of a fuzzy logic controller the membership functions and the rule base are potentional candidates for evolution [16]. In this approach we consider a fixed rule base as it was created by the human expert that designed the fuzzy logic controller and we evolve the membership functions with a genetic algorithm.

The chromosome is created by encoding the real values of the numbers that define the membership functions of the input and output variables as described in fig. 1.

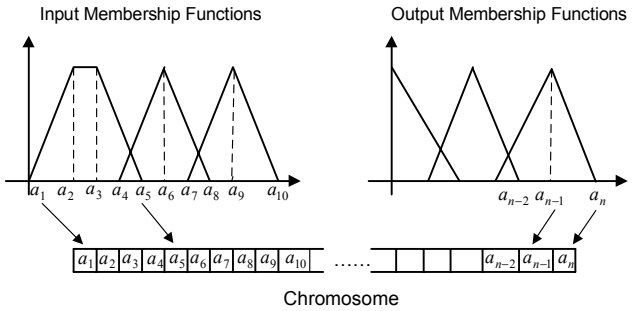


Fig. 1 Chromosome created by the membership functions

Each input and output variable is encoded as an array $Cin_i$ where $i=1,...,$ *number_of_input_variables*, and each output variable is encoded as $Cout_j$ where $j=1,...,$ *number_of_output_variables*. The overall chromosome has the following form:

$$C=[Cin_i\ Cout_j]. \quad (7)$$

The length of the chromosome is related to the number and the type of the membership functions of the input and output variables and to their number. An initial population is created and each individual, which is a fuzzy logic controller, of a single generation is tested in the same experiment. After the completion of the experiments the performance of each individual is evaluated based in the fitness function used, and the individuals are ranked. For the selection process a roulette wheel with slots according to fitness is used, as described in [17]. After that the crossover is performed and mutation of the final population. In all the experiments performed the variables of the genetic algorithm like the crossover and the mutation probability are the same. This has as a result all the variations in the behaviors of different controllers to attribute to the different type of fitness functions.

## IV. CUSTOM ROBOT VEHICLE

There is a lot of argument about using simulation or experiments in real robots for the evolution of their controllers. Some researchers propose the use of simulators to initially evolve the controllers and then validate them in real robots. We believe that although simulation has proven to be a useful tool for the evolution of robot controllers, the evolution in real robot can incorporate factors that a simulation, no matter how accurate it is, cannot consider. For this reason we have designed and developed a low cost robot that allows to experimentate and validate our approach.

All the work presented in this manuscript has been done on a real robot. We will briefly describe the robot's parts as long as the essential software developed for control and sensor processing.

The basic part of the mobile robot is consisted by the

*Rogue Blue* educational robot. Several modifications were made and new sensors and devices were added since the minimal configuration of the initial platform wasn't suitable for experimentation. The sensor suite of the robot is consisted from a sonar range device positioned in the center in the upper front part and two infrared sensors positioned in the lower left and right part of the vehicle. It has two odometers one in each wheel and an electronic compass in the upper center part. The two wheels of the robot are rotating from one servo motor each. The low level control of the servo motors and the data acquisition from the sensors and the other devices is performed from a board equipped with an OOPic microprocessor. The robot has a Bluetooth device which allows the communication of the robot with a host computer that has a Bluetooth base station attached. The robot is also equipped with a device from *Digital Solutions* that allows inter-robot communication without the intervention of a base station in cases that the experiments require more than one robot. The configuration of the devices with which the robot is equipped is presented in fig. 2.

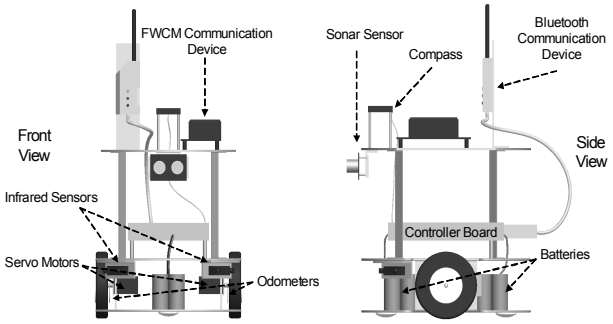


Fig. 2 Configuration of the devices attached in the custom robotic vehicle

The OOPIC board is running a program which allows the low level control of the devices and data acquisition from the sensors. In a base station a MATLAB session is running and it's exchanging data with the robot. The position calculation, the fuzzy logic controller and the genetic algorithm are running in the base station. It should be noted that although the configuration is a host / slave system, conceptually the robot is considered as an autonomous agent.

For the calculation of the robot's position the data from the odometers are used. The position of the robot is calculated based on the equations presented in [13] for this specific type of robot.

## V. EXPERIMENTAL RESULTS AND ANALYSIS

The experiments conducted in an area with maximum width of 3.7 meters and maximum length of 4.5 meters. The floor was covered with carpet, in order to minimize the sliding effect. The experimentation area and the real robot are presented in fig. 3.

For each fitness function type, we evolved the controllers for 80 generations, using the same experimental set up. In each generation the goal of the robot was to navigate from an initial point to a final target point. The environment is static with obstacles in predefined positions. Each individual had 30 seconds to accomplish its goal and after that the experiment was terminated and the robot was repositioned.

Fig. 3 Experimental testbed

An additional parameter was added to the fitness functions mainly because the experiments were performed in a real robot. This parameter takes into account the activation level of each sensor. In case that the mean activation level is less than 0.3 (experimentally derived) a penalty value was assigned to the fitness functions of the specific individual. That was done to avoid cases in which the robot has stuck to an obstacle and therefore could not complete the experiment. Obviously all individuals who fall in this category have limited chances to survive.

The evolved controllers were tested for various navigation conditions and scenarios. Four different sets of experiments were conducted, and each individual had one minute to go as close as possible to a target point. Several parameters concerning the robot movement were monitored. Some of them are shown in Table I for the best individuals of each type of fitness function. The trajectories followed by the best individuals, in certain test cases are presented in Fig. 4 – 7.

TABLE I
EXPERIMENTAL RESULTS

| Test Case 1 | *Aggregate* | *Behavioral* | *Tailored* |
|---|---|---|---|
| Minimum Distance (mm) | 63,8980 | 65,4110 | 47,8566 |
| Time (sec) | 27,9322 | 30,5126 | 27,7254 |
| Mean Sensor Activation | 0,8744 | 0,8642 | 0,8799 |
| Turn Rate | 3,7337 | 3,1252 | 2,7086 |
| Mean Lin. Vel. (cm/sec) | 9,39702 | 8,90375 | 9,42322 |
| Test Case 2 | | | |
| Minimum Distance (mm) | 41,8505 | 59,1457 | 42,6037 |
| Time (sec) | 42,7507 | 30,5774 | 42,1305 |
| Mean Sensor Activation | 0,8926 | 0,8031 | 0,8880 |
| Turn Rate | 3,5242 | 3,6257 | 3,6288 |
| Mean Lin. Vel. (cm/sec) | 9,21670 | 9,00040 | 9,19336 |
| Test Case 3 | | | |
| Minimum Distance (mm) | 41,8505 | 59,1457 | 42,6037 |
| Time (sec) | 42,7507 | 30,5774 | 42,1305 |
| Mean Sensor Activation | 0,8926 | 0,8031 | 0,8880 |
| Turn Rate | 3,5242 | 3,6257 | 3,6288 |
| Mean Lin. Vel. (cm/sec) | 9,21670 | 9,00040 | 9,19336 |
| Test Case 4 | | | |
| Minimum Distance (mm) | 125,3659 | 110,7722 | 122,1138 |
| Time (sec) | 41,07354 | 40,2912 | 44,62946 |
| Mean Sensor Activation | 0,816438 | 0,81505 | 0,82027 |
| Turn Rate | 8,042438 | 7,78897 | 7,21267 |
| Mean Lin. Vel. (cm/sec) | 8,254323 | 8,152524 | 7,998842 |

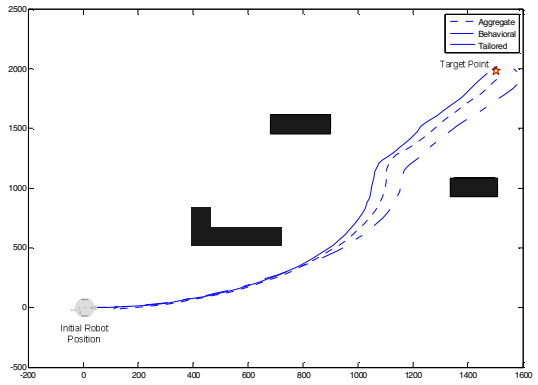


Fig. 4 Test Case 1

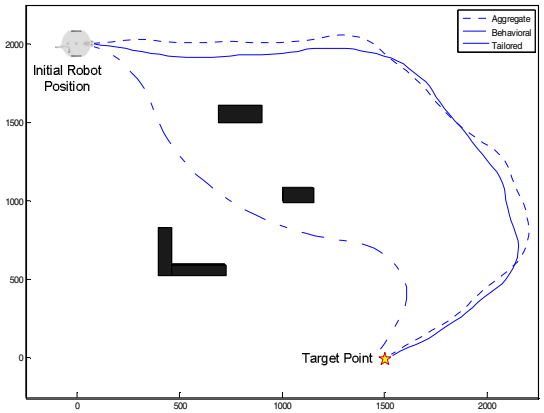


Fig. 5 Test Case 2

In order to measure the overall behavior of the controllers we introduce a metric of the efficiency of the robot's performance. The efficiency metric is given by:

$$EF(i)=\frac{\min\{MD_C\}}{MD_i}+\frac{\min\{T_C\}}{T_i}+\frac{MSA_i}{\max\{MSA_C\}}+\frac{MLV_i}{\max\{MLV_C\}} \quad (8),$$

where, *MD* is minimum distance to target, *MST* is the time needed for the minimum distance, *MSA* is the mean sensor activation, *MLV* is the mean linear velocity, $c=1,\ldots i$ is the group of robot controllers considered. The performance of each fitness type for all test cases is presented in Table II.

TABLE II
EFFICIENCY OF THE FITNESS FUNCTION TYPE FOR EACH TEST CASE

| | Test Case | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| *EF* | Aggregate | 3,732 | 3,715 | 3,837 | 3,859 |
| | Behavioral | 3,567 | 3,583 | 3,816 | 3,981 |
| | Tailored | 4 | 3,700 | 3,681 | 3,778 |

In order to identify which fitness is affecting certain attributes of the robots performance, we conduct the following analysis. We calculate the mean *EF* of each function in all test cases by excluding in each case one factor i.e minimum distance from target.

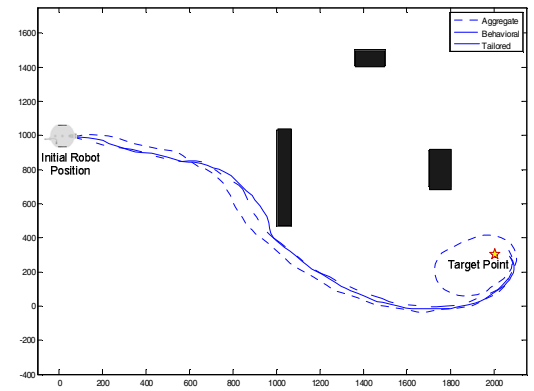


Fig. 6 Test Case 3

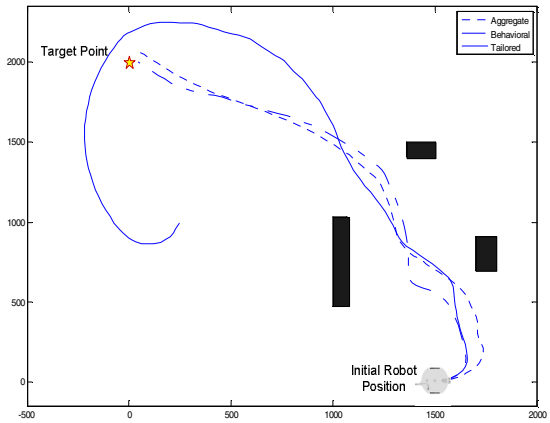


Fig. 7 Test Case 4

The cases that we examine are:

$$\text{Case 1: } \overline{EF(i)-\frac{\min\{MD_C\}}{MD_i}}, \quad (9)$$

$$\text{Case 2: } \overline{EF(i)-\frac{\min\{T_C\}}{T_i}}, \quad (10)$$

$$\text{Case 3: } \overline{EF(i)-\frac{MSA_i}{\max\{MSA_C\}}}, \quad (11)$$

$$\text{Case 4: } \overline{EF(i)-\frac{MLV_i}{\max\{MLV_C\}}}, \quad (12)$$

The results are presented in Table III.

TABLE III
EFFICIENCY FACTOR'S ANALYSIS

| | Aggregate | Behavioral | Tailored |
|---|---|---|---|
| *EF* | 3,786 | 3,737 | 3,790 |
| Case 1 | 2,878 | 2,918 | 2,836 |
| Case 2 | 2,905 | 2,760 | 2,930 |
| Case 3 | 2,789 | 2,769 | 2,793 |
| Case 4 | 2,787 | 2,764 | 2,812 |

In all cases except case 1, the tailored fitness function outperforms the aggregated which is followed by the

behavioral. In genera, it seems that the most suitable fitness for this task is the tailored one. The case which behavioral outperforms the others was case 1. This is probably due to the fact that there was no variable measuring the distance, from the target in the behavioral fitness. The other two types of functions had this kind of metric which led the evolution process in a more target oriented.

## VI. DISCUSSIONS AND CONCLUSIONS

Fuzzy logic controllers without appropriate tuning represent a pure reactive solution for the mobile robot navigation problem. Therefore, evolution processes have been extensively used to optimize the structural characteristics (mainly membership functions and rules) of fuzzy controllers. In these cases, the performance of the evolved controller is heavily based on the selected fitness function.

In this paper we examine the impact of the selection of the fitness function on the evolution of a fuzzy logic controller, together with the navigation performance of a real robotic vehicle. We used different types of fitness functions, namely, *aggregate*, *behavior* and *tailored*, that represent different evolution approaches. In order to evaluate the performance of the evolved controllers and investigate how fitness functions affect the overall behavior of the vehicle we introduced the efficiency factor metric. From the experiments conducted, it turned out that the "tailored" type function is more appropriate for the navigation problem in static environments.

Future research will include the formulation of a metric for the measurement of the efficiency of the performance of a team of robot vehicles. We also anticipate studying the effects of fitness function selection on controllers that avoid dynamically moving obstacles.

## VII. ACKNOWLEDGMENT

L. Doitsidis was supported from "Herakleitos" fellowships for research from the Technical University of Crete EPEAEK II 88727. The authors would like to thank S. Piperidis and C. Anastasopoulos for their help.

## REFERENCES

[1] N. C. Tsourveloudis, L. Doitsidis, K. P. Valavanis, "Autonomous Navigation of Unmanned Vehicles: A Fuzzy Logic Perspective," in *Cutting Edge Robotics,* V. Kordic, A. Lazinica, M. Merdan, Eds. Mammendorf, Germany: Pro Literatur Verlag, 2005, pp. 291-310.

[2] X. Yang, M. Moallem, R. V. Patel, "A layered Goal-Oriented Fuzzy Motion Planning Strategy for Mobile Robot Navigation," *IEEE Trans. Syst., Man, Cybern., B,* vol. 35, no. 6, pp. 1214-1224, Dec. 2005.

[3] P. Resu, E. M. Petriu, T. M. Whalen, A. Cornell, H. J. W. Spoelder, "Behavior-Based Neuro-Fuzzy Controller for Mobile Robot Navigation," *IEEE Trans. Instrum. Meas.,* vol. 52, no. 4, pp. 1335-1340, Aug. 2003.

[4] N. C. Tsourveloudis, K. P. Valavanis, T. Hebert, "Autonomous Vehicle Navigation Utilizing Electrostatic Potentional Fields and Fuzzy Logic," *IEEE Trans. Robotic. Autom.* vol. 17, no. 4, pp. 490-497, Aug. 2001.

[5] C. Ye, N. H. C. Yung, D. Wang, "A Fuzzy Controller With Supervised Learning Assisted Reinforcement Learning Algorithm for Obstacle Avoidance," *IEEE Trans. Syst., Man, Cybern., B,* vol. 33, no. 1 pp. 17-27, Feb. 2003.

[6] S. –I. Lee, S. –B. Cho, "Emergent Behaviors of a Fuzzy Sensory-Motor Controller Evolved by Genetic Algorithm," *IEEE Trans. Syst., Man, Cybern., B*, vol. 31, pp. 919-929, Dec. 2001.

[7] S. H. Kim, C. Park, F. Harashima, "A Self-Organized Fuzzy Controller for Wheeled Mobile Robot Using an Evolutionary Algorithm," *IEEE Trans. Ind., Electron.,* vol. 48, no. 2, pp. 467-474, Apr. 2001.

[8] H. Hagras, V. Callaghan, M. Colley, "Learning and adaptation of an intelligent mobile robot navigator operating in unstructured environment based on a novel online Fuzzy-Genetic system," *Fuzzy Set. Syst.,* vol. 141, pp. 107-160, 2004.

[9] F. Hoffman, G. Pfister, "Evolutionary Design of a Fuzzy Knowledge Base for a Mobile Robot," *Int. J. Approx. Reason.,* vol. 17, no. 4, pp. 447-469, 1997.

[10] V. Matellan, C. Fernadez, J. M. Molina, "Genetic Learning of Fuzzy Reactive Controllers," *Robot Auton. Syst.,* vol. 25, pp. 33-41, 1998.

[11] D. P. T. Nanayakkara, K. Watanabe, K. Kiguchi, K. Izumi, "Evolutionary Learning of a Fuzzy Behavior Based Controller for a Nonholonomic Mobile Robot in a Class of Dynamical Environments," *J. Intell. Robot. Syst.,* vol. 32, pp. 255-277, 2001.

[12] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines.* Cambridge, Massachusetts: MIT Press, 2000.

[13] J. Borenstein, H. R. Everett, L. Feng, "Where Am I? Sensors and Methods for Mobile Robot Positioning," Univ. Michigan, Ann Arbor, MI, 1996.

[14] L. Doitsidis, A. L. Nelson, M. T. Long, K. P. Valavanis, R. R. Murphy, "Experimental Validation of a MATLAB Based Control Architecture for Mobile Robot Outdoor Navigation," In *Proc. 13th IEEE Med. Conf. Control Automation*, Limassol, Cyprus, 2005, pp. 1499-1505.

[15] D. Floreano, F. Mondana, "Evolution of homing navigation in a real mobile robot," *IEEE Trans. Syst., Man, Cybern., B,* vol. 26, no. 3, pp. 396-407, Jun. 1996.

[16] O. Cordon, F. Herrera, F. Hoffmann, L. Magdalena, *Genetic Fuzzy Systems: Evolutionary Tuning and Learning of Fuzzy Knowledge Bases.* Singapore: World Scientific, 2001.

[17] Z. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*. Heidelberg: Springer-Verlag, 1994.