Automatic lateral control for unmanned vehicles via genetic algorithms☆

E. Onieva[a,*], J.E. Naranjo[b], V. Milanés[a], J. Alonso[a], R. García[a], J. Pérez[a]

[a] Industrial Computer Science Department, Centro de Automática y Robótica (UPM-CSIC), La Poveda-Arganda del Rey, 28500 Madrid, Spain
[b] Department of Intelligent Systems, Polytechnic University of Madrid (UPM), 28031 Madrid, Spain

ABSTRACT

It is known that the techniques under the topic of Soft Computing have a strong capability of learning and cognition, as well as a good tolerance to uncertainty and imprecision. Due to these properties they can be applied successfully to Intelligent Vehicle Systems; ITS is a broad range of technologies and techniques that hold answers to many transportation problems. The unmanned control of the steering wheel of a vehicle is one of the most important challenges facing researchers in this area. This paper presents a method to adjust automatically a fuzzy controller to manage the steering wheel of a mass-produced vehicle; to reach it, information about the car state while a human driver is handling the car is taken and used to adjust, via iterative genetic algorithms an appropriated fuzzy controller. To evaluate the obtained controllers, it will be considered the performance obtained in the track following task, as well as the smoothness of the driving carried out.

Keywords:
Autonomous vehicles
Genetic algorithms
Fuzzy control
Intelligent Transportation Systems (ITSs)
Lateral control

1. Introduction

Intelligent Transportation Systems (ITS) apply information and communication techniques in order to achieve safe and efficient driving. In the automotive industry, ITS mainly is used to provide information to the driver and, in some cases, they are connected to the car actuators, attempting to minimize injuries and to prevent collisions [1].

The work described in this paper has been carried out at the AUTOPIA Program of the Centre of Automatic and Robotic, a part of the Spanish Council for Scientific Research (CSIC), which is focused on the area of autonomous vehicles. More concretely, the aim of the present work is the automatic control of the steering wheel (also known as lateral control) of a mass-produced vehicle, although another area of the research carried out by the AUTOPIA Program is longitudinal control, the control of the vehicle's speed and its adaptation to road features, using the throttle and the brake pedal as needed [2,3]. Research projects similar to AUTOPIA exist in different countries [4–7].

Vehicles used by the AUTOPIA program are equipped with instrumentation and software necessary to perform their autonomous management, since an autonomous car must control some or all of its functions without external intervention [8–10]. To carry out the present work, they have been required: An asphalt set of test roads called ZOCO (acronym for Driving Zone), differential GPS coverage supplied by a high-precision GPS base station and a wireless LAN for differential correction generation and transmission and a mass-produced Citron C3 Pluriel equipped with a DGPS, actuators for the steering wheel and pedals, and a computer connected to the communications network. There is long way until an implementation of full automatic-steering control comes on the market [11]. Due to that, researchers around the entire world focus their efforts in the study, implementation and validation of this kind of systems as is showed in the famous driverless cars competition sponsored by the Defense Advanced Research Projects Agency[1] (DARPA), the central research organization of the United States Department of Defense. DARPA has sponsored three competitions in the area of autonomous vehicles.

Classical control techniques are the usual way to manage complex systems such as the steering wheel of a car [12,13]. Other way is the use of artificial intelligence techniques; these methods are specially indicated when we try to emulate human control actions, such as human car driving. Fuzzy logic [14] has become a particularly widely used methodological approach to these tasks since Sugeno's work on fuzzy control in 1985 and 1987 [15,16]. Fuzzy

[1] http://www.darpa.mil/.

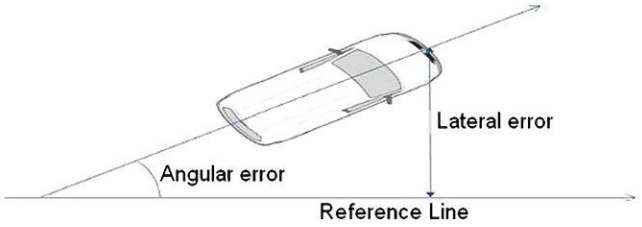


Fig. 1. Input variables graphical representation.

systems arose from the desire to describe complex systems linguistically, and fuzzy controllers allow a human approach to control design without the demand for knowledge of mathematical modeling of more conventional control design methods [17]. Usually, generation of the membership functions and rule base that define a fuzzy controller is a task mainly done either iteratively, by trial-and-error, or by expert knowledge. A task like this one is a natural candidate to be solved by means automatic techniques from the artificial intelligence field.

Genetic algorithms (GA) are general purpose search algorithms which use principles inspired by natural genetic populations to evolve solutions to problems [18,19]. The basic idea is to maintain a population of chromosomes, which represent candidate solutions that evolves over time through a process of competition and controlled variation. Each chromosome in the population has an associated fitness to determine which chromosomes are used to form new ones in the competition process, which is called selection. The new ones are created using genetic operators such as crossover and mutation. GA have been used in literature in task of learn or optimize fuzzy controllers [20–22].

The ORBEX [23] (Spanish acronym for Fuzzy Experimental Computer) fuzzy development environment has been used for the definition and implementation of fuzzy controllers. With ORBEX, several ways of driving can be defined to emulate different types of drivers: calm, quick, brusque, etc., or to adapt the driving to traffic conditions: platoons, overtaking, etc. These strategies can be defined and implemented by means of *if ... then ...* rules in almost natural language [24].

This paper approaches the topic of using of genetic algorithms in the design of fuzzy logic controllers a real world application: a mass-produced vehicle steering wheel management. After a first part of capture of information about a human driver handling, that information is processed in order to get relevant information about the attitude of the driver. Once done this, a system able to get the information and return an appropriate fuzzy controller has been created, via application of an iterative genetic algorithm. The GA has been implemented in two iterative phases, the first one will improve the membership functions and the second one the rule base of the fuzzy controller. Finally, obtained controllers have been tested in a private and experimental area to verify that they show a similar behavior to the shown by human driver.

2. Information capture and processing

Generated fuzzy controllers will use two inputs obtained from the match between the GPS positioning information and the reference route, defined as GPS digital cartography. The input variables are: the *lateral error* and *angular error*. The lateral error represents the distance of the current car position to the theoretical car position if it was on the desired trajectory, the reference line; its values can take any value $(-\infty, +\infty)$. The angular error is the angle shaped by the reference line and the car velocity vector; its values are restricted to the interval $(-180, 180)$. In Fig. 1 it can be seen the graphical representation of these values.

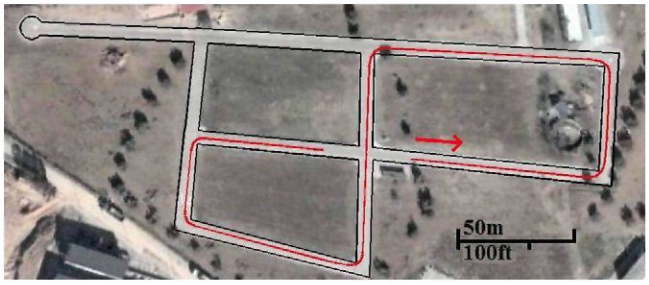


Fig. 2. GPS reference route over an aerial view of ZOCO.

The output will be the reference position sent to a low-level control layer consisting of a PID controller that manages a motor attached to the steering bar to take it to the reference position.

2.1. Information capture

The GPS maps that specify the desired route are built automatically by tracking the route with a GPS-equipped car and, after the end of the run, a computer system selects the most significant waypoints that will be used as a reference line to measure lateral and angular errors and human drivers' actions. The reference line created to capture information on the human drivers' behavior is shown in Fig. 2 over an aerial view.[2] The route consists of three curves towards the left followed by three curves towards the right. Once the desired route had been built, two human drivers followed it. Then, a large data set was obtained for processing and, after that, a set of input and output values to apply the genetic algorithm is obtained. The data set contains information on the lateral and angular errors of the car and the human drivers' actions.

2.2. Information processing

The process starts with the data set that compiles the human driver's actions; this data set is showed in Fig. 3 (left), where the lateral and angular errors are presented on the X–Y axis, and the position of the steering wheel adopted by the human driver on the Z-axis, between -1 (maximum turn to the right) and 1 (maximum turn to the left).

The large number of points obtained almost provides a specific control surface. This also can slow down the process for estimating the fuzzy controller because the resulting controller response has to be compared with the human response. These responses will therefore make some empirical values that are added to the data set insignificant. Thus, there must be some way of guaranteeing freedom to the controller design and speed to the process. Accordingly, in first place, the values of the set of points are normalized in the $[-1, 1]$, assuming a limit for the lateral error of ± 5 m. After, a 21×21 grid is defined on the X–Y plane, and at each point in the grid the mean output of the closest points in the real input is taken. As can be seen in the graphs, there are zones in the grid without any associated value, so human driver actions in extreme situations, like an angular error of $\pm 180°$, ... are not reflected in the graph. To assure that some extreme cases are covered, the following points deduced from common sense are added to the point swarms:

- (1, 1, 1): if the lateral error is maximum to the right and the angular error is maximum to the right, steering must be maximum to the left;

[2] http://maps.google.com/.

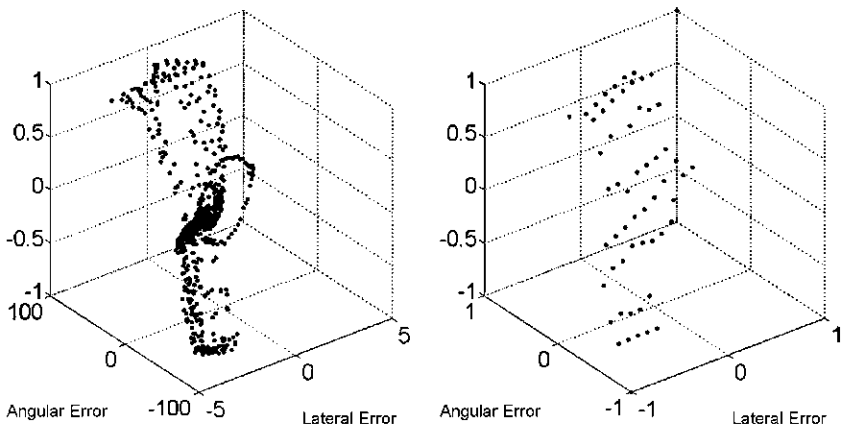


Fig. 3. Initial Data Set (left). Data set used to estimate the fuzzy controller (right).

- $(-1, -1, -1)$: if the lateral error is maximum to the left and the angular error is maximum to the left, steering must be maximum to the right.

Finally, in Fig. 3 (right) the final sets of points can be seen which are used as inputs to the genetic algorithm to ensure similarity with the human behavior observed and freedom to generate fuzzy controllers with a smooth surface. This means that the car will maintain the same orientation without any brusque steering movements.

3. Fuzzy controller optimization

It is natural to think that the fuzzy controller for managing the steering wheel of a car must be symmetric; this would mean that the action taken when the inputs have a certain value must be inverse to the action taken when the inputs have inverse values. However, it is not the case because the aim of this work is to reproduce human driving so the controller does not have to be symmetric. Therefore, the system must be able to distinguish, for example, a driver who drives too far to the right of the reference line (for example, if he is driving a very slow vehicle, he must go to the right so as not to disturb the other drivers).

A Mamdani fuzzy system is used to model the controller. The controller will have two fuzzy inputs, called Lateral Error and Angular Error. The fuzzyfication of each input variable is done by means of four or five membership functions (Section 3.1). There is only 1 fuzzy output variable, called Steering, with 21 linguistic labels, whose membership functions are defined by singletons. The minimum has been used for defining the tnorm (*AND*) and the maximum for the tconorm (*OR*). In this paper, it will be distinguished three different rule bases of different complexity, explained in Section 3.2. The defuzzyfication method applied is Center of Mass.

The representation of the fuzzy controllers with which the genetic algorithm will work is divided into two different parts: on the one hand, the representation of the membership functions (MF) and, on the other, the representation of the rule base (RB). This allows the process to work with and to improve both aspects of the fuzzy controller independently.

3.1. *Membership functions representation*

To generate the membership functions, two ways has been used; one of them to define a controller with 4 MF for each input variable and other one for a controller with 5. To represent 4 MF, 12 values are used $(x_1, x_2, \ldots, x_{12})$. The first six represent two membership functions for the linguistic variables high/low left deviation (HLD and LLD) and the last six represent two membership functions for the low/high right deviation (HRD and LRD), as it can be seen in Fig. 4 (top). To represent 5 MF 16 values are used $(x_1, x_2, \ldots, x_{16})$. Again, the first six represent two membership functions for the linguistic variables high/low left deviation (HLD and LLD), the next four represent the membership function no deviation (NO), and the last six represent two membership functions for the low/high right deviation (HRD and LRD); as we can see in Fig. 4 (bottom).

To ensure that the membership functions have coherent values, the method will have to preserve some constraints, following ones are applied when a controller with 4 MF is defined:

(1) $(-1 < x_{1,2,3,5,6,7} < 0)\,(0 < x_{8,9,10,11,12,4} < 1)$.
(2) $(x_5 < x_6)\,(x_1 < x_2 < x_3 < x_4)\,(x_7 < x_8 < x_9 < x_{10})\,(x_{11} < x_{12})$.
(3) $(x_5 < x_2)\,(x_9 < x_{12})$.
(4) $(x_1 < x_6)\,(x_{11} < x_{10})$.
(5) $(x_1 < x_7)\,(x_4 < x_{10})$.

And the next ones are used when a controller with 5 MF is defined:

(1) $(-1 < x_{1,2,3,4,5,6,7,8} < 0)\,(0 < x_{9,10,11,12,13,14,15,16} < 1)$.
(2) $(x_5 < x_6)\quad (x1 < x_2 < x_3 < x_4)\quad (x_7 < x_8 < x_9 < x_{10})\quad (x_{11} < x_{12} < x_{13} < x_{14})\,(x_{15} < x_{16})$.
(3) $(x_5 < x_2)\,(x_3 < x_8)\,(x_9 < x_{12})\,(x_{13} < x_{16})$.
(4) $(x_1 < x_6)\,(x_7 < x_4)\,(x_{11} < x_{10})\,(x_{15} < x_{14})$.
(5) $(x_1 < x_7)\,(x_6 < x_4)\,(x_{10} < x_{14})\,(x_{11} < x_{15})$.

In both lists, restriction sets (1) guarantees that the error towards the left is represented by negative values and towards the right by positive values; restriction sets (2) guarantees that the membership functions are trapeziums; restriction sets (3) guarantees that every input value is covered with a maximum degree by, at the most, one membership function; restriction sets (4) guarantees that every input value is covered by at least, one membership function; and, finally, restriction sets (5) guarantees logical membership functions and avoids illogical values.

The output can be one of the 21 Sugeno's singletons showed in Fig. 5, which is equivalent to a type I representation (with trapeziums) and is better for a control purpose because it allows faster operations [25].

The singletons are uniformly distributed in the interval $[-1, 1]$, with a separation between two adjacent ones of 0.1. The *R* denotes that the steering turn is in the right direction, and the *L* that it is in the left direction, and the number represent the strength of the turn; for example, *R10* means a full turn of the steering wheel to the right, *L5* a half turn to the left. *N* means no turn of the steering wheel. The singletons for the output of the controller do not

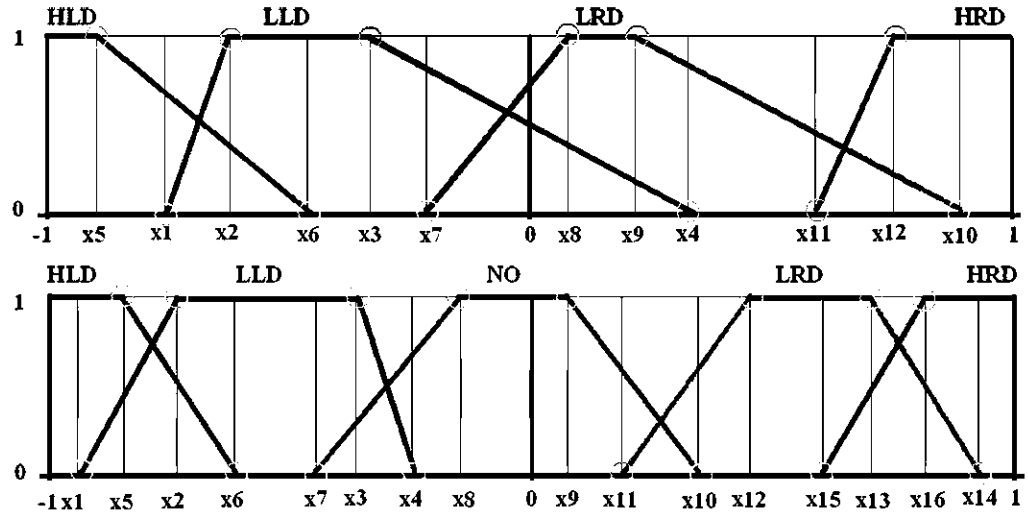


Fig. 4. (Top) Representation of 4 MF with 12 values. (Bottom) Representation of 5 MF with 16 values.

Table 1
Cases covered by the central rule base (C) and the marginal rule base (M). Label *NO* is not used in a 4 MF controller representation.

Lat\Ang	∅	HLD	LLD	{NO}	LRD	HRD
∅		M	M	M	M	M
HLD	M	C	C	C	C	C
LLD	M	C	C	C	C	C
{NO}	M	C	C	C	C	C
LRD	M	C	C	C	C	C
HRD	M	C	C	C	C	C

Table 2
Number of rules for each RB/MF possible combination.

RB\MF	4	5
Marginal	8	10
Central	16	25
Total	24	35

change, the genetic algorithm will be in charge of assigning them as consequents of predefined rule bases presented in next subsection.

3.2. Rule base representation

For the fuzzy controller, a set of three RB are available; the GA will work with the consequents of the rules, changing the consequent assigned to one rule to obtain the best configuration of rules for the controller, i.e., for each possible antecedent, there will be a rule. Irrespective of whether the controller uses 4 or 5 MF for each input variable, there are 3 different rule bases called *Marginal*, *Central* and *Total*, respectively. The Marginal RB works only with rules with a simple antecedent; the Central RB represents the rules with AND-composed antecedents; the Total RB consists of the union of the Central and Marginal RB. Cases covered by each RB can be seen in Table 1, where each cell (X, Y) in the table represents a rule like IF (Angular is X) AND (Lateral is Y) THEN Steering is ST, where ST can be one of the 21 singletons described previously. Table 2 shows the total number of rules of each possible controller representation, in function of the number of MF and the kind of RB used.

As in the case of optimization of the membership functions, the rule base also contains specific restrictions in order to improve the application of the method to the problem. Mainly, the restriction that the rule base has to fulfill for each i and j rule is that if the antecedent of rule i is greater or equal to that of the antecedent of rule j, the singleton used in rule i has to be greater or equal to that of the antecedent used in rule j. To represent the rule base in a way that can be used by the genetic algorithm, an integer coding is used to represent the singleton assigned to the rule. Thus, a vector of the integers between 1 and 21 is needed with as many elements as rules as the rule base used and the vector will be optimized by means of an integer-coded genetic algorithm.

3.3. Iterative genetic optimization schema

The method is implemented in two different phases, which are repeated a given number of times and represent two GA; one of them in charge of improving MF and the other one the RB; the general schema used is showed in Fig. 6.

The GA phases, both over the MF and RB, follows a steady-state genetic algorithm eschema, where, in each generation two individuals are selected to apply over them the crossover operator to generate two offsprings, then, they are mutated and inserted (or not) again in the population. As comented in previous subsections, a two-row real matrix with 12 or 16 columns is used to represent the membership functions of the controllers, and a vector of 8, 10, 16, 24, 25 or 35 integer elements is used to represent the rule base. the GA has been implemented by following the following operators:

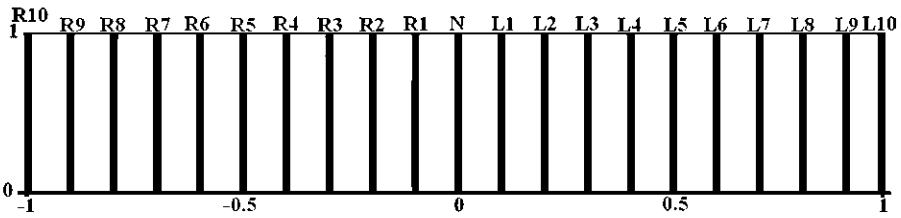


Fig. 5. Membership functions for the output.

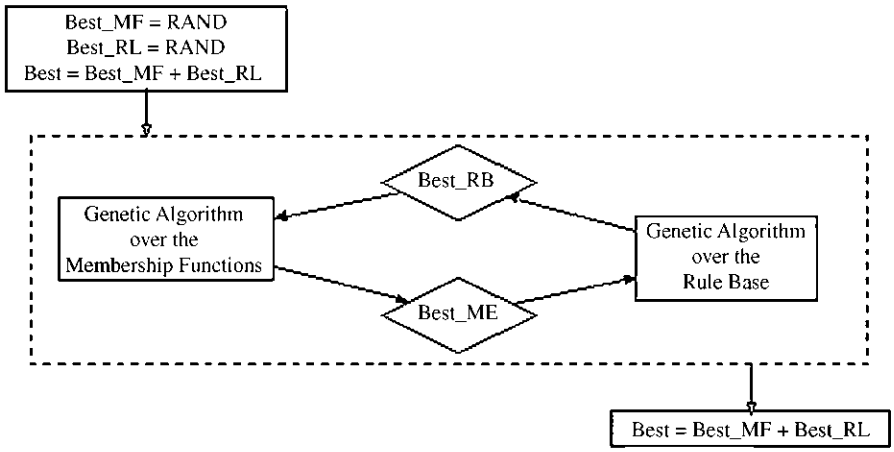


Fig. 6. Method schema used.

- *Selection*: in each generation, two chromosomes are selected to apply the crossover operator on them. They are selected by binary tournament.
- *Crossover operator*: BLX-α [26] applied to every value of the representation makes it possible to generate two new chromosomes from two parent chromosomes in the optimization phase of the membership functions and a simple crossover [11] for the optimization of the rule base.
- *Mutation operator*: each position of every generated chromosome undergoes a random change according to a probability defined by a mutation rate, the mutation probability, p_m [18,19].
- *Replacing method*: each new chromosome generated will replace the worst chromosome in the population if its fitness is better.

Each genetic algorithm will work with a population size of N_{MF} or N_{RB} and will be executed during G_{MF} or G_{RB} generations, respectively.

To measure the quality (*fitness*) of a given controller two factors must be considered: adjustment to human driver actions and to provide a smooth surface in its central zone; this zone is already the most used. To evaluate the first factor, the mean square error *MSE* is used between the controller output and the human drivers' actions. To evaluate the second factor, the surface is evaluated in a grid (i,j), $i,j = [-1, -0.9, \ldots, 0, \ldots, 0.9, 1]$ and D is taken to be the largest difference between two adjacent points in the grid. The fitness function used will be the weighted aggregation of *MSE* and D ($F = v \cdot MSE + (1-v) \cdot D$), with $v = 0.75$ (so the similarity with human behavior is given more importance than the smooth surface). The genetic algorithm will try to minimize this value.

4. Experimentation and results

In the presented method we can observe the following aspects that have to be considered when parameters for the experiments are configured: there is always a copy of the best chromosome found in the population; before the improvement of each part of the controller (MF and RB) a new population is created and the improvement of each part of the fuzzy controller is done separately. Bearing in mind the characteristics of the method, the parameters must fulfill the following premises:

- $N_{MF/RB}$ must be small because there is always a copy of the best chromosome found, and the probability of the chromosome being chosen from the best will be greater to explore the chromosomes near to it.
- $G_{MF/RB}$ must also be small because optimization of both parts of the controller is done separately and many iterations of the method with few generations are preferable to few iterations with many generations.
- *It* (number of iterations) must be large because of the previous comment.
- The α parameter used in the BLX-α crossover must be small ($\alpha \leq 0.5$), so the exploitation of the chromosomes in the population is given more importance than the exploration of new chromosomes (this part is obtained by two separate genetic algorithms).
- The mutation probability p_m must be high to change a large number of values of a chromosome.

Bearing in mind these considerations, the parameters used for executing the method are: $\alpha = 0.2$, $p_m = 0.25$, $It = 100$, $N_{MF} = N_{RB} = 10$, $G_{MF} = G_{RB} = 20$.

Once processed the input data in order to obtain a representative set of human's actions on the steering wheel (Section 2), the iterative genetic method (Section 3) has been applied to obtain a set of fuzzy controllers representing the human drivers' behavior. Fig. 7 shows the control surfaces of the controllers obtained, where it can be seen that all of them perform the extreme case restrictions, a full steering turn to the left if the lateral and angular error is maximum to the right and a full steering turn to the right if the lateral and angular error is maximum to the left.

Controller has been tested in the Driving Zone using the GPS reference route shown in Fig. 2. The behavior of each fuzzy controller is shown next. Notice that all the routes were done at a velocity of between 15 km/h in curved lines and 25 km/h in straight lines. The experiments are shown by means of North-East UTM coordinates, where the reference route and the route followed by the autonomous car are represented with each controller obtained. The routes followed by the car with each controller obtained are presented in Fig. 8.

In Fig. 8, a relative good behavior of the controllers can be observed, it can be seen some differences between some controllers; most of them are able to track a straight segment of the road, but all present the same failure, i.e., after a curve, the car moves away from the reference line. This failure may be because of the relatively high speed used to circulate around the test zone; or probably, because of the non-predictive model of the controllers, so that once the car leaves the curve, it does not know where the next target point is and keeps the steering wheel turn until the next GPS target position is processed (post-curve movement outside the reference line). To compare the ability of each controller to follow the route, the lateral and angular error obtained along the track

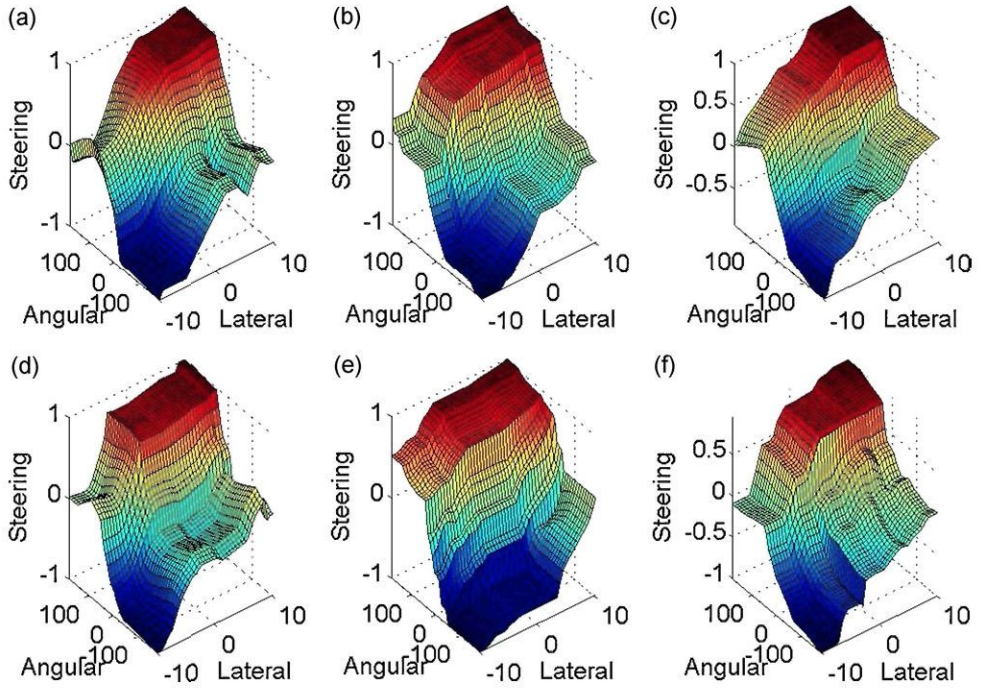


Fig. 7. Control surfaces of the controllers obtained: 4 MF + Marginal RB (a), 4 MF + Central RB (b), 4 MF + Total RB (c), 5 MF + Marginal RB (d), 5 MF + Central RB (e), 5 MF + Total RB (f).

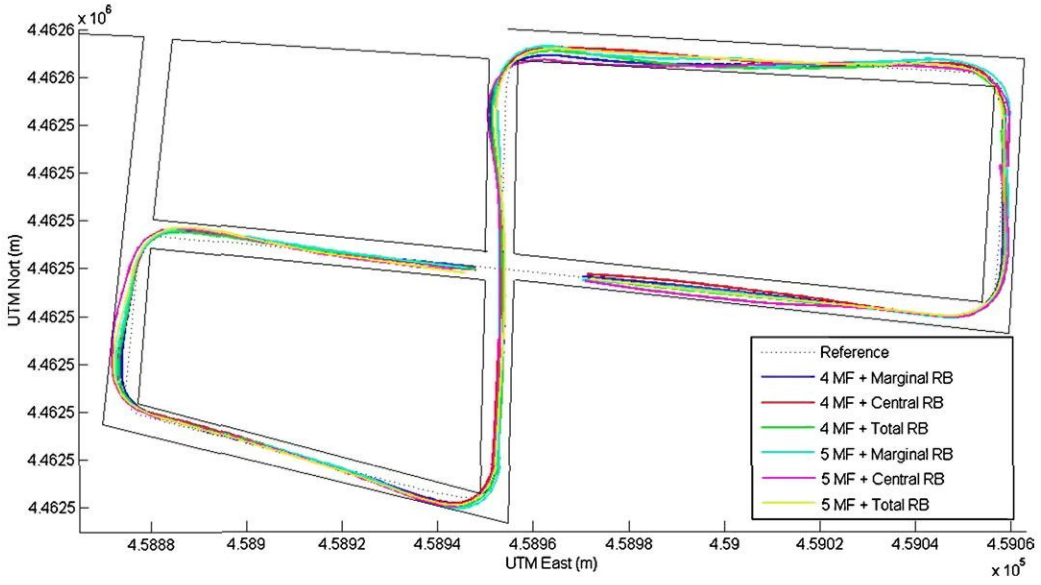
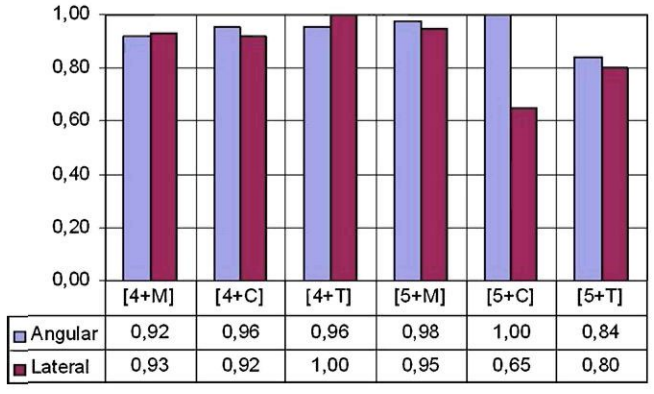


Fig. 8. Route followed by obtained fuzzy controllers.



	[4+M]	[4+C]	[4+T]	[5+M]	[5+C]	[5+T]
Angular	0,92	0,96	0,96	0,98	1,00	0,84
Lateral	0,93	0,92	1,00	0,95	0,65	0,80

Fig. 9. Comparison graphic.

has been saved, and, Fig. 9 shows the normalized average of them, where it can be seen more precisely the differences between the behavior of each controller.

Even the routes seem to be very similar, Fig. 9 shows that the worst behavior for the angular error was obtained by controller [5 MF + Central RB] and the worst behavior for the lateral error by [4 MF + Central RB]; there is not a notable difference between the controllers that use 4 membership functions, even the [5 MF + Marginal RB] controller presents a values very similar to the controllers with 4 MFs. The bigger difference can be found in the controller [5 MF + Central RB] and [5 MF + Total RB]; [5 MF + Central RB] presents the minimum lateral error mean but, as has been said, the maximum angular error. The minimum angular error value can be seen in controller [5 MF + Total RB] and it presents an acceptable lateral error, so it is natural to think that [5 MF + Total RB] can be selected as the best controller obtained.

5. Conclusions and future works

This work has presented an automatic adjustment of fuzzy controllers designed to automatically manage the steering wheel of a mass-produced vehicle which was equipped with the necessary instrumentation and software. To achieve this, an iterative genetic algorithm based method was implemented, capable of iteratively adjusting the membership functions and rule base which define a fuzzy controller. The method applied genetic algorithms with some constraints applied to the controllers to guarantee that the results could perform the automatic driving of an unmanned car. The controllers also had to have a smooth surface in order to guarantee safe and comfortable driving for the car occupants. The controllers were tested on a private asphalt circuit, and most of them showed good behavior in straight lines, but also bad behavior in curve lines; this is caused by the non-predictive model used in the control, with no analysis of the non-immediate reference points of the desired route. The controllers showed very smooth driving, even when they were circulating at relatively high speeds. This is a good starting point for subsequent research in ITSs. Subsequent research will focus on improving the models obtained here in order to guarantee precise driving and also comfortable and smooth driving along a reference line. Furthermore, the consequences of using a determined number of membership functions will be studied and any other car state variable, rule base, ... Safety and efficiency must also not be forgotten. All these important features will be considered in future research.

References

[1] W.D. Jones, Keeping cars from crashing, IEEE Spectrum (September) (2001) 40–45.

[2] J.E. Naranjo, C. González, J. Reviejo, R. García, T. de Pedro, Adaptive fuzzy control for inter-vehicle gap keeping, Special Issue on Adaptive Cruise Control, IEEE Transactions on Intelligent Transportation Systems 4 (3) (2003) 132–142.

[3] V. Milanés, J.E. Naranjo, C. González, J. Alonso, T. de Pedro, Autonomous vehicle based in cooperative GPS and inertial systems, Robotica 26 (2008) 627–633.

[4] M. Bursa, Big names invest heavily in advanced ITS technology, ISATA Magazine (2000) 24–30.

[5] J.M. Blosseville, M. Parent, The French program: La route automatisée, IEEE Intelligent Systems (2000) 10–13.

[6] Literature Review of ADAS/AVG (AVCSS) for Japan and United States, 2001.

[7] R. Bishop, Intelligent vehicle applications worldwide, IEEE Intelligent Systems (2000) 78–81.

[8] J.E. Naranjo, C. González, J. Reviejo, R. García, T. de Pedro, M.A. Sotelo, Using fuzzy logic in automated vehicle control, IEEE Intelligent Systems 22 (1) (2007) 36–45.

[9] R. García, T. de Pedro, Automatic car drivers, ISBN 0 9 532 576 0 6, in: Presented at the Proc. 31st Int. Symposium Automotive Technology and Automation, June, 1998.

[10] M.A. Sotelo, S. Alcalde, J. Reviejo, J.E. Naranjo, R. García, T. de Pedro, C. González, Vehicle fuzzy driving based on DGPS and vision, in: Proc. 9th IFSA World Congress, 2001.

[11] E. Dickmanns, The development of machine vision for road vehicles in the last decade, in: Proc. IEEE Intelligent Vehicles Symposium, 2002, pp. 268–281.

[12] R. Marino, S. Scalzi, G. Orlando, M. Netto, A nested PID steering control for lane keeping in vision based autonomous vehicles, in: Proc. American Control Conference, 2009, pp. 2885–2890.

[13] J. Ackermann, W. Sienel, Robust control for automated steering, in: Proc. of the 1990 American Control Conference, 1990, pp. 795–800.

[14] L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.

[15] T. Takagi, M. Sugeno, Fuzzy identification of systems and its applications to modeling and control, IEEE Transactions on Systems, Man and Cybernetics SMC-15 (1) (1985) 116–132.

[16] M. Sugeno, T. Murofushi, T. Mori, T. Tatematsu, J. Tanaka, Fuzzy algorithmic control of a model car by oral instructions, in: K. Hirota, T. Yamakawa (Eds.), IFSA'87 Special Issue on Fuzzy Control, 1987.

[17] S. Chaib, M.S. Netto, S. Mammar, Hinfin; adaptive, PID and fuzzy control: a comparison of controllers for vehicle lane keeping, in: Proc. IEEE Intelligent Vehicles Symposium, 2004, pp. 139–144.

[18] J.H. Holland, Adaptation in natural and artificial systems, in: The University of Michigan Press, 1975.

[19] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison Wesley, New York, 1989.

[20] F. Herrera, M. Lozano, J.L. Verdegay, Tuning fuzzy logic controllers by genetic algorithms, International Journal of Approximate Reasoning 12 (1995) 299–315.

[21] A. Homaifar, E. McCormick, Simultaneous design of membership functions and rule sets for fuzzy controller using genetic algorithms, IEEE Transactions on Fuzzy Systems 3 (2) (1995) 129–139.

[22] M. Mucientes, D.L. Moreno, A. Bugarín, S. Barro, Design of a fuzzy controller in mobile robotics using genetic algorithms, Applied Soft Computing 7 (2) (2007) 540–546.

[23] R. García, T. de Pedro, Modeling a fuzzy coprocessor and its programming language, Mathware and Soft Computing 5 (2–3) (1998) 167–174.

[24] R. García, T. de Pedro, First application of the ORBEX coprocessor: control of unmanned vehicles, in: EUSFLAT-ESTYLF Joint Conference, Mathware and Soft Computing 7 (2–3), 2000, pp. 265–273.

[25] M. Sugeno, On stability of fuzzy systems expressed by fuzzy rules with singleton consequents, IEEE Transactions on Fuzzy Systems 7 (1999) 201–224.

[26] L.J. Eshelman, J.D. Schaffer, Real coded genetic algorithms and interval schemata, in: L. Darrell Whitley (Ed.), Foundation of Genetic Algorithms 2, Morgan Kaufmann Publishers, San Mateo, 1993, pp. 187–202.