Optimization of an Autonomous Car Fuzzy Control System via Genetic Algorithms

Javier Alonso Ruiz, Nacho Serrano, Teresa de Pedro, Carlos González and Ricardo García
Industrial Automation Institute, CSIC (Higher Scientific Research Center)
La Poveda, Arganda del Rey, 28500 Madrid, Spain.
E-mail: {jalonso, nachosm, tere, gonzalez, ricardo }@iai.csic.es

***Abstract*— We have built a fuzzy autonomous car control system for navigating bends as smoothly as possible. To get the optimal parameters of the membership functions of the fuzzy control system, we have built a genetic algorithm application that runs our simulator thousands of times, minimizing the effort made on steering, and maintaining an acceptable path error. We have used a high mutant rate genetic algorithm (that we call radioactive environment evolution) with a fitness function having two objectives. The results are impressive: the steering efforts when the vehicle is entering the bend are drastically reduced and the steering correction efforts inside the bends have disappeared. Although we have achieved such good results, it is important to remember that they are simulator based. We are now testing this control on real autonomous cars with relatively good results. However, we hope to achieve better real results as we improve our simulation program.**

I. Introduction

One of the chief concerns in a control system is to reduce the force applied on the actuators. In autonomous vehicles the amount of electricity required is an important issue. Our goal is to reduce the electrical consumption in the direction actuators. Moreover, the reduction of excessive steering wheel movements increases the feeling of comfort. We have proposed and tested a control method for vehicle navigation in bend sections. We have used a curvature control method based on fuzzy logic with genetic algorithms optimization on a geometric trajectory design. We have achieved impressive results in simulations, and we are now testing these results on real autonomous vehicles.

We have used a fuzzy logic control system to get a solution that is as smooth as possible [2]. This control system must maintain the curvature goal that depends on the point of the bend at which the car is located. It will work on the curvature error, its first instant derivation and the error accumulated on that bend, just to minimize the curvature error. Therefore, we use a PI fuzzy control.

The membership function parameters are improved using a genetic algorithm fitness function that combines the accumulated effort and the maximum error made on the circuit bends. We have calculated different optimized set values depending on the type of the circuit bends (motorways with small curvature or race circuits).

The simulator uses detailed maps of competition circuits, mainly Xerez, maintaining real proportions. And it takes into account the mechanical features of our CITROËN cars. We have carried out several simulations on this circuit, and the results show a drastic reduction in effort after the genetic algorithm optimization.

II. Trajectory Design

To navigate the car within the smoothest possible trajectory, it is designed in the same way as motorway bends are built. We get the curvature radius and the maximum permitted speed for each bend. We use this data to design a bend whose curvature increases from zero (straight line) to its maximum value at the start and decreases to zero (the next straight line) at the end. The curvature increase and decrease ratios are set by a parameter that depends on the maximum speed.

Therefore, for this designed path, the car's steering will move slowly from zero (wheels in the straight position) to the angle required for bend navigation and then back to zero at the end.

III. Fuzzy Controller

The control system is mainly based on differential GPS. It gets a high precision GPS position for the car and compares it with a designed path map. It then sets the goals for the steering wheel, the accelerator and the brake. We chose a fuzzy control system whose aim is to avoid big changes in the driving goals (steering wheel, accelerator and brake goals).

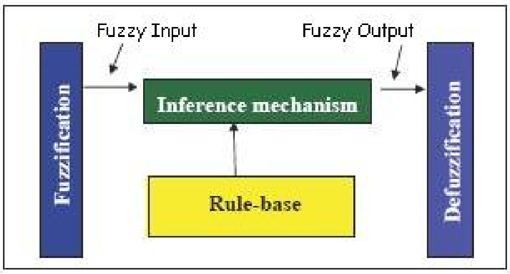

Fig. 1. Fuzzy Controller Diagram

To explain the fuzzy control system, we will describe it step by step.

3.1 Fuzzification

First of all, the discrete values from sensors are converted into membership degrees of input fuzzy linguistic variables. Each linguistic variable has one or more fuzzy partitions, and these partitions are defined by a membership function. To turn a sensor discrete value into n membership degrees for the n fuzzy partitions of the fuzzy linguistic variables, we have:

$$\mu_{Partition\_n}(x) \in [0,1] \qquad x \in X \qquad \textbf{(1)}$$

Where x is the sensor's discrete value and X is the sensor's range of possible values.

The input fuzzy linguistic variables will have only one fuzzy partition, but we will use the fuzzy modifiers LESS THAN and MORE THAN for operations, as we have three fuzzy partitions. His fuzzy membership functions will be very simple, just to reduce the computational cost in a real time real car controller.

The state of the system is described with different sets of variables depending on car behavior. If the car is tracking a straight line of reference, the variables are:

1) *Lateral error:* the distance between the center of the back axis of the car and the line of reference (in meters).
2) *Angular error:* the difference between the car's direction and the line of reference direction (in degrees).

Hence, the fuzzy partitions will be *Lateral_ error* ∊ {zero} and *Angular_ error* ∊ {zero}, and its fuzzy membership functions will be:

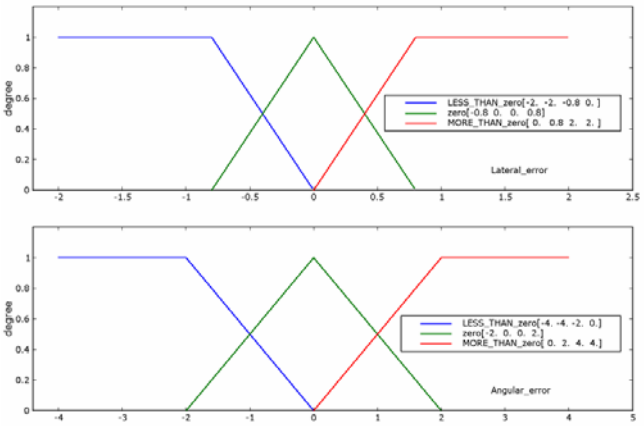

Fig. 2. The three membership functions of "Lateral error" and "Angular error" linguistic variables

If the car is tracking a bend, the variables are based on the curvature. The car's curvature is calculated from the steering wheel position and the distance between front and rear axes of car. These variables are:

3) *Curvature error:* the difference between the car's desired and real curvature.
4) *Curvature error summation:* the accumulated curvature error.

Then, the fuzzy partitions will be *Curvature_error* ∊ {zero} and *Sum_Curvature_error* ∊ {zero}, and its fuzzy membership functions will be:

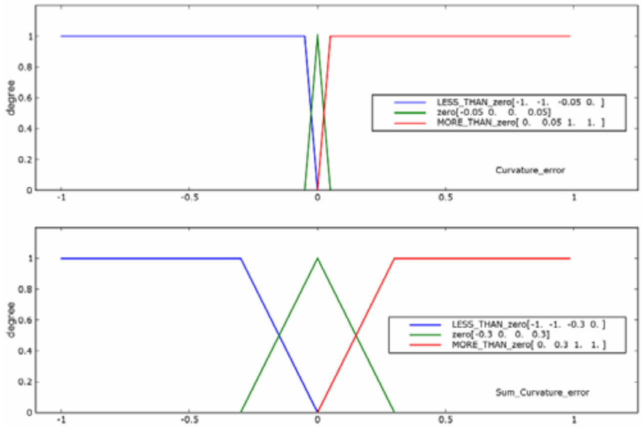

Fig. 3. The three membership functions of "Curvature error" and "Curvature error summation" linguistic variables

For both behaviors, the speed is controlled by the following variables:

5) *Speed error:* the difference between desired and real speed.
6) *Acceleration:* The speed variation over time.

Now, the fuzzy partitions are *Speed_ error* ∊ {zero} and *Acceleration* ∊ {zero}, and its fuzzy membership functions are:

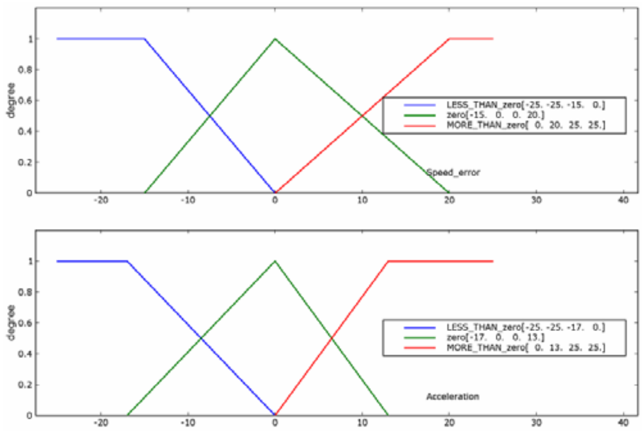

Fig. 4. The three membership functions of "Speed error" and "Acceleration" linguistic variables

3.2 Inference Mechanism

The inference mechanism yields the fuzzy output from the fuzzy input using the IF-THEN rules from the Rule Base. It is divided in two steps:

1. *Matching:* In this step, the fuzzy input variables are used to calculate the truth value of the premises. For each rule, all the antecedent variables are computed to get the truth value of the premise. We have chosen the MIN/MAX inference method. Therefore, the operators for computing all the antecedent variables are:

Classic Operator	Fuzzy equivalent
AND	$\mu_{A \wedge B} = \min\{\mu_A, \mu_B\}$
OR	$\mu_{A \vee B} = \max\{\mu_A, \mu_B\}$
NOT	$\mu_{\neg A} = 1 - \mu_A$

2. *Inference:* For each output variable in the consequent, its fuzzy set will be modified by the actual truth values of its premise. This variable is modified as follows:

Chop off the top: $\mu_{rule(n)}(u)=\min\{\mu_{premise}(n), \mu_{action}(u)\}$

Where $\mu_{rule(n)}(u)$ is the u partition fuzzy set modified by the rule n and its premise, $\mu_{premise}(n)$ is the truth value of the premise n, and $\mu_{action}(u)$ is the u partition of the above fuzzy set.

3.3 Rule Base

We use two different fuzzy rule bases depending on the car's desired behavior. The first one is used on the straight-line sections of the circuit, and the second one will be used on the bend sections.

The fuzzy rules used in both fuzzy rule bases are:

Rule	IF	*Speed _ error*	THEN	Accelerator
1	IF	MORE THAN (zero)	THEN	restrain
2	IF	zero	THEN	maintain
3	IF	LESS THAN (zero)	THEN	accelerate

Rule	IF	*Acceleration*	THEN	Accelerator
4	IF	MORE THAN (zero)	THEN	restrain
5	IF	zero	THEN	maintain
6	IF	LESS THAN (zero)	THEN	accelerate

The fuzzy rules for the straight-line sections are:

Rule	IF	*Lateral_ error*	THEN	Steering
7	IF	MORE THAN (zero)	THEN	posSmall
8	IF	zero	THEN	maintain
9	IF	LESS THAN (zero)	THEN	negSmall

Rule	IF	*Angular_ error*	THEN	Steering
10	IF	MORE THAN (zero)	THEN	posSmall
11	IF	zero	THEN	maintain
12	IF	LESS THAN (zero)	THEN	negSmall

The fuzzy rules for the bend sections are:

Rule	IF	*Curvature_ error*	THEN	Steering
7	IF	MORE THAN (zero)	THEN	negative
8	IF	zero	THEN	maintain
9	IF	LESS THAN (zero)	THEN	positive

Rule	IF	*Sum_ Curvature_ error*	THEN	Steering
10	IF	MORE THAN (zero)	THEN	negative
11	IF	zero	THEN	maintain
12	IF	LESS THAN (zero)	THEN	positive

The fuzzy operators used on the rules are defined as:

Operator	Fuzzy equivalent
LESS THAN	$\exists y \in A \mid \mu_A(y) = 1 \rightarrow$ $\left(\forall x \in A, x < y \rightarrow \mu_{<A} = 1 - \mu_A\right)$ $\wedge \left(\forall x \in A, x \geq y \rightarrow \mu_{<A} = 0\right)$
MORE THAN	$\exists y \in A \mid \mu_A(y) = 1 \rightarrow$ $\left(\forall x \in A, x < y \rightarrow \mu_{<A} = 0\right)$ $\wedge \left(\forall x \in A, x \geq y \rightarrow \mu_{<A} = 1 - \mu_A\right)$

3.4 Defuzzification:

The actuation over the system is defined by a goal position for the steering wheel and another goal position for the accelerator-brake set. The output variables are normalized, steering wheel range goes from -540 to 540 degrees and accelerator-brake set goes from -27.1 to 16.3 Km/h/s [6].

The fuzzy partitions of the fuzzy linguistic variables are *Accelerator* ∈ {restrain, maintain, accelerate} and *Steering* ∈ {negative, negSmall, maintain, posSmall, positive}, and its fuzzy membership functions are:

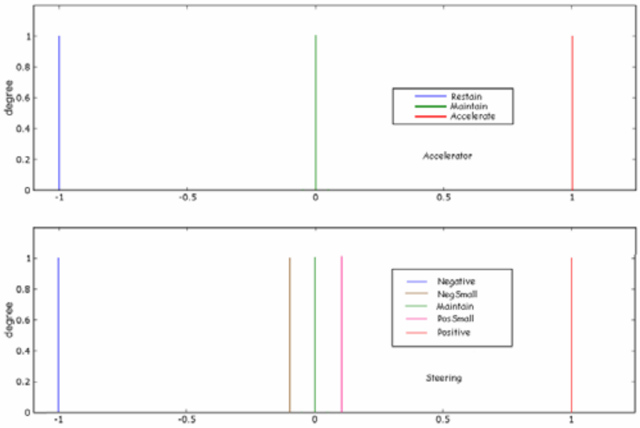

Fig. 5. Membership functions of "Accelerator" and "Steering" linguistic variables

Once the inference mechanism has finished, each fuzzy partition of each fuzzy output linguistic variable will have its own membership degree. We will use the gravity center method to defuzzify each variable.

$$Output = \frac{\sum_i b_i \int \mu_{rule\_i}}{\sum_i \int \mu_{rule\_i}} \quad \textbf{(2)}$$

$$\int \mu_{rule\_i} \quad \textbf{(3)}$$

Where bi is commonly the center of each fuzzy partition, but, in this case, is just the crisp number that defines each fuzzy partition, and the integral expression (3) is the area of each fuzzy partition after inference, which is, in this case, just the membership degree of each partition.

IV. Initial Results of the Fuzzy Controller

The simulator uses detailed maps of competition circuits, mainly Xerez, with its real proportions. It takes care of the mechanical characteristics of our cars, CITROËN. And it also takes care of the sample period of the GPS.

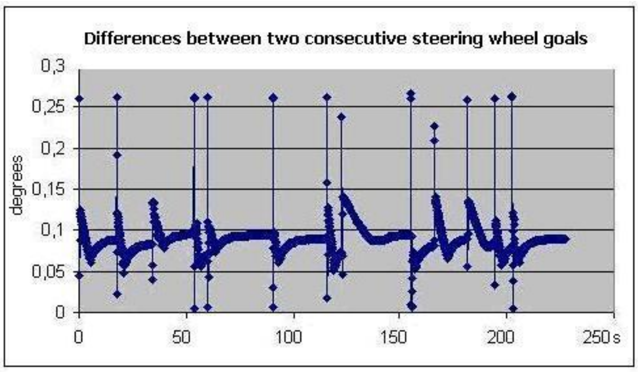

Fig. 6. Differences between consecutive steering goals

The figure shows the efforts made on bend sections only. These are the initial results of the car around the simulated Xerez circuit with the fuzzy controller and the manually fitted fuzzy membership functions. The trajectory over the marked path is good enough, but the difference between consecutive steering goals is higher than expected.

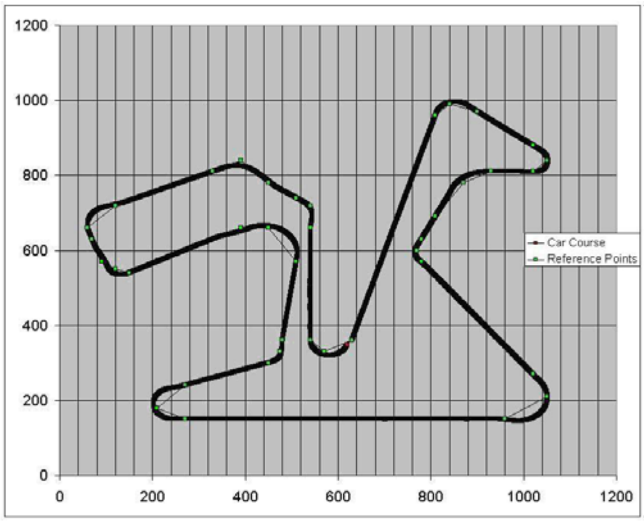

Fig. 7. Car trajectory on the Xerez Circuit path

In the figure above, the car course (red dots, sights as a black line) goes through almost all the reference points (green points). And it's always adjusted to the circuit dimensions.

V. Fuzzy Control Optimization

To improve the efficiency of the system, we will use genetic algorithms as an optimization technique.

5.1 Optimization fitness function

To get a smooth control as possible, we try to minimize the steering wheel movements. Therefore, we define effort as the difference between consecutive steering wheel goals. To ensure that the car trajectory still matches the car's designed path, we add the maximum error on bend sections to the fitness function. This error is the maximum of the minimum distances between the car and the reference points on the bend sections. We combine both objectives in the fitness function:

Fitness = k1 * effort + k2 * Bends_maximum_error **(4)**

We have observed in previous simulations that the best effort is not less than 200 degrees, and the maximum error is from 15 to 20 meters (maximum distance to an intermediate blend point). Then, based on the previous observations, we chose constant values of k1=1 and k2=20 to attempt to give both objectives equal priority.

5.2 Genetic Coding

The fuzzy control system behavior depends on the membership functions of the input and output variables, the rules, and the inference method. But, in this case, we chose to modify only the values that define the fuzzy membership functions of the input and output variables. The input fuzzy membership functions are symmetrical triangular functions, whose highest value (1) is on zero. Hence, they can be defined by only one parameter. This parameter is codified with 8 bits, and the value range is 0 to 255. We divide each parameter by 100 to get a precision of 0.01 and a range of 0 to 2.55.

We will leave the variables for navigating straight-line sections unmodified, because they are well tested [3]. Hence, only the parameters that define the fuzzy membership functions of the variables that are used for bend sections control are codified. But we do not codify negative and negSmall parameters, because they will be obtained by changing the positive and posSmall sign.

$Curvature\_\ error \in \{\text{zero}\}$
$Sum\_\ Curvature\_\ error \in \{\text{zero}\}$
$Steering \in \{\text{posSmall, positive}\}$

We also codify the minimum curve radius of the designed path. This parameter determines the tightest allowed bend on the designed path, and it's used to design the trajectory on bend sections [1]. This parameter is also codified with 8 bits that means 0.1 precision and a value range of 0 to 25.5 meters. Then our population of individuals has 5 genes that are codified in 40 bits.

So, the first four genes: "Curvature Error symmetrical triangular membership functions positive limit", "Curvature Error Summation zero positive limit", "Steering positive small singleton output" and "Steering positive singleton output" are codified with 8 bits and have a value range of 0 to 2.55 with a precision of 0.01. The fifth gene is the minimum curve radius parameter, also codified with 8 bits, a value range of 0 to 25.5, and a 0.1 precision.

5.3 Optimization Algorithm

The population size has been changing due the experimental results until the results were good enough. At first, we used a population of 10 individuals, which we tested over 10,000 generations, but the genome of all the individuals turned out to be the same. Therefore, we increased the population to 100 individuals and reduced the generations to 200.

5.4 Initialization

The whole population was created from two individuals called progenitors, which performed well in previous simulations. As we wanted to guide the search we have choose to make a small variation in each new individual modifying one bit from a random gene (mutation) to get each new individual seemed as his progenitor. Forty-nine new individuals were created in this way from each progenitor, and, at the end, we got a population of 100 individuals.

5.5 Evaluation

For the purposes of evaluation, each individual was tested in a virtual Xerez circuit simulation. Each gene was decoded and the membership functions of the fuzzy controller changed. After simulation, the fitness value obtained was written down, and the program continued with the next individual.

5.6 Selection

First, the best individual is kept as the elite individual. Then, the program selects 70 individuals for crossover operations to create the next generation of individuals. Any individual can be chosen for crossover, but each one has a different probability of being selected.

$$\text{Probability}[\text{i}] = \frac{1/\ fitness[i]}{1/\sum_{i=0}^{99} fitness[i]} \qquad (5)$$

From the population that has not been selected, the elite individual replaces the worst individual and the last 29 unmodified individuals are mutated at a rate of 20%. This high mutation rate has been selected to increase the diversity of the population, and delay the convergence to the last generations. However the mutation operator may modify an individual to make it better, or, in contrast, make it worst. The former is the usual behavior in this application. So we have chosen this selection method after several experiments to guarantee that the crossed individuals aren't mutated.

5.7 Crossover and Mutation

In the crossover procedure, a random point between the genes is chosen for each pair of individuals. The first new individual will have the genes of one parent up to the crossover point and the genes of the other as of the crossover point. And the second one will have the rest.

The mutation procedure randomly affects one bit of the individual. The bit changes from zero to one or vice versa.

5.8 Final Selection

Once the 200 generations have been evaluated, the best individual is chosen and its parameters decoded.

VI. RESULTS AFTER OPTIMIZATION

The car trajectory after genetic algorithm optimization was just as good as the initial one.

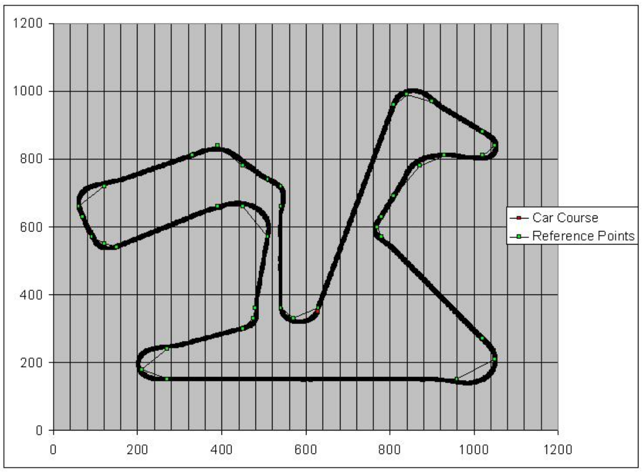

Fig. 8. Car trajectory on the Xerez Circuit path after optimization

In this figure the car course goes through almost all the reference points, as figure 7. But the distance with those reference points are slightly better.

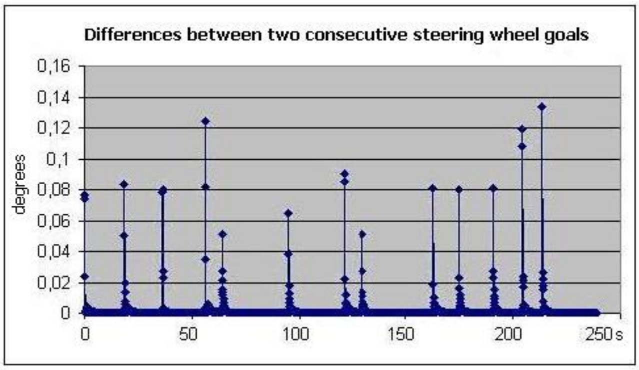

Fig. 9. Differences between consecutive steering goals after simulation

The differences between consecutive steering goals are drastically reduced. Fig. 9 shows the results. The bigger goal differences appear only at the beginning and at the end of each bend (note that effort is only measured on the bend sections, and the starting efforts for a bend are shown after the ending efforts for the previous bend). And within the bend navigation, the steering wheel remains unchanged.

VII. CONCLUSIONS

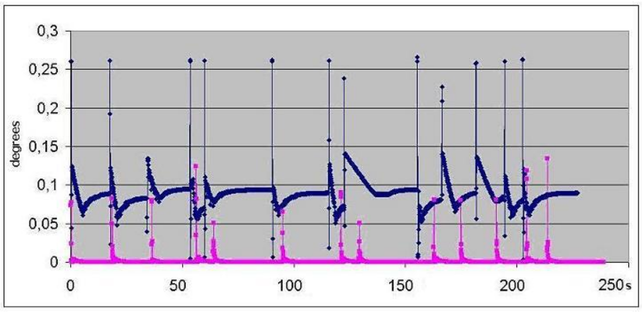

Fig. 10 Differences between two consecutive steering wheel actuator goals. The blue and the pink lines are the steering wheel actuation before and after the application of the genetic algorithm, respectively.

This figure shows the effort made on bend sections (that we have defined as the difference between two consecutive steering wheel actuator goals). The upper function show the results with manually fitted membership functions for the fuzzy control variables (also shown on figure 6) and the lower function show the effort results after the GA optimization (shown on figure 9).

As shown in Figure 10, the efforts on bend sections are drastically reduced, as much in the middle of each bend (figure flat sections) as at the starting and ending points (high value points).

We are now testing this control on real vehicles with relatively good results. However, we hope to achieve better real results as we improve our simulation program.

ACKNOWLEDGMENT

This work was supported by the project ISAAC, DPI2002-04064-C05-02 Spanish Ministerio de Educación y Ciencia and project COPOS, Ministerio de Fomento BOE 280, November 22, Res. 22778.

REFERENCES

[1] Alonso Ruiz J, de Pedro T, González C and García R, "Soft Computing and Geometrical Control for Computer Aided Driving" Eurocast 2005: Tenth International Conference on Computer Aided Systems Theory

[2] García R and de Pedro T, "First Application of the ORBEXoprocessor : Control of Unmanned Vehicles" *Mathware and Soft Computing*, no. 7, vol 2-3, pp 265-273, 2000

[3] García R, de Pedro T, Earanjo J E, Reviejo J and González C, "Frontal and Lateral Control for Unmanned Vehicles in Urban Tracks" *IEEE Intelligent Vehicle Symposium (IV2002)*

[4] Jiménez Shaw J, "Resolución numérica completa de la Ecuación de la Clotoide", web page and Clothos constants calculation program. http://javier.jimenezshaw.com/

[5] Laugier C and Fraichar T, "Decisional Architectures for Motion Autonomy" chapter 11 in *Intelligent Vehicle Technologies*, editors: Vlacit, ParenT, Harashima, SAE International, ISBN 0 7680 0780 I

[6] Naranjo J E, Reviejo J, Gonzalez C, García R, de Pedro T, "A Throttle & Brake Fuzzy Controler: Towards the Automatic Car" Lecture Notes in Computer Science: Computer Aided Systems Theory – Eurocast 2003, LNCS 2809 pp 291-301

[7] Web page www.carreteros.org, Norm 3.1-IC "Trazado"