On-Line Learning of a Fuzzy Controller for a Precise Vehicle Cruise Control System

E. Onieva[a], J. Godoy[b], J. Villagrá[b], V. Milanés[c], J. Pérez[d]

[a] *Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain. email:* `enrique.onieva@decsai.ugr.es`

[b] *AUTOPIA program at the Centro de Automática y Robótica (UPM-CSIC), La Poveda-Arganda del Rey, 28500 Madrid, Spain. email:* `{jorge.villagra, jorge.godoy}@csic.es`

[c] *California PATH, University of California at Berkeley, Richmond, CA 94804-4698, USA. email:* `vicente.milanes@berkeley.edu`

[d] *IMARA team at INRIA research center, Paris - ROCQUENCOURT. France. email:* `joshue.perez_rastelli@inria.fr`

Abstract

Usually, vehicle applications need to use artificial intelligence techniques to implement control strategies able to deal with the noise in the signals provided by sensors, or with the impossibility of having full knowledge of the dynamics of a vehicle (engine state, wheel pressure, or occupants' weight).

This work presents a cruise control system which is able to manage the pedals of a vehicle at low speeds. In this context, small changes in the vehicle or road conditions can occur unpredictably. To solve this problem, a method is proposed to allow the on-line evolution of a zero-order TSK fuzzy controller to adapt its behaviour to uncertain road or vehicle dynamics.

Starting from a very simple or even empty configuration, the consequents of the rules are adapted in real time, while the membership functions used to codify the input variables are modified after a certain period of time. Extensive experimentation in both simulated and real vehicles showed the method to be both fast and precise, even when compared with a human driver.

Keywords: Intelligent Transportation Systems, Autonomous Vehicles, Fuzzy Control, On-Line Learning, Speed Control.

1. Introduction

Intelligent Transportation Systems (ITS) constitute a broad range of technologies applied to transportation to make systems safer, more efficient, more reliable, and more environmentally friendly, without necessarily having to physically alter existing infrastructure [1]. In the automotive industry, sensors are mainly used to give the driver information. In some cases, they are connected to a computer that performs certain control actions such as attempting to avoid collisions and, if unavoidable, to minimize injuries [2].

Autonomous vehicle guidance represents one of the most important challenges of ITS. It involves two different controls, one associated with the steering wheel, termed *lateral* [3] control, and the other associated with the control pedals (and in some cases the gear shift) [4].

Excessive or inappropriate speed is one of the main causes of traffic accidents [5]. That is one of the main reasons why automatic speed control is presently one of the most popular research topics throughout the automotive industry. The goal of this automation is to improve safety by relieving the human drivers of tasks that could distract their attention, as well as making the traffic flow more efficient.

There are different approaches to speed regulation. Cruise control (CC) systems have the capability of maintaining a pre-set speed. Adaptive Cruise Control (ACC) systems add the capability of maintaining a safe distance from a preceding vehicle [6] by using information coming from on-board devices. Other approaches are ACC with communications (CACC) [7, 8] which incorporates the capability of interchanges of information between cars so as to improve performance and safety, or ACC with Stop

& Go capability (SGACC) [9] to manage situations in which the car must be stopped. Automation of both the throttle and the brake pedals is needed before installing these features in a vehicle.

Some manufacturers incorporate CC or ACC systems in their cars, but in many cases they do not operate at low speeds. These systems have been widely studied in the specialist literature, usually in simulated environments [10, 11, 12]. The focus, both in industry and in academic research, has generally been on application to highway driving [13, 14]. The reason that low-speed contexts have generally not been considered is that actions on the pedals more strongly affect the car's dynamics [15] making the system hard to model, simulate, or control. In urban environments, it is quite usual that the speed must be reduced and then kept low even when there is no vehicle in front due, for example, to the presence of school zones where time must be allowed to react to unpredictable or other sudden events (a pedestrian crossing in front of the car or a traffic light turning red). Indeed, the typical speed limit in urban environments is 50 km/h, for which the various forms of CC speed management systems are inappropriate.

The objective of this work is to create a system capable of allowing the evolution of fuzzy rules for the management of the pedals of a vehicle in urban driving contexts. The use of fuzzy logic [16] for control systems has two main advantages. (i) Fuzzy logic obviates the need to use complex approximate models that are either computationally inefficient if they are realistic, or unrealistic if they are computationally efficient. (ii) The aim is not to represent the system mathematically, but to emulate the behaviour and experience of human drivers. There is no systematic approach to the design of fuzzy controllers [17]. Instead, how they are designed depends on the knowledge available about the system to be controlled.

The system's evolution must be on-line in order for the controller to adapt to changing road or vehicle conditions such as slopes, gear changes, weight of the occupants, or other unpredictable parameters. To this end, one defines a zeroth-order TSK fuzzy controller [18] with trapezia for codifying inputs and singletons as consequents. An initial fuzzy controller with all consequents located at zero (with the meaning that the pedals are not acted upon) evolves over time, adapting both the position of the singletons and the granularity of the trapezia.

For the initial empty controller to evolve, a first module is designed that adapts the positions of the singletons defining the consequents of the system depending on the speed and the acceleration of the vehicle. After a certain amount of time, a second, structural learning module takes responsibility for adding or modifying the trapezia that codify the input variables of the system. Finally, a third module is in charge of filtering the pedal actions, with the aim of emulating human actions.

One line of work on on-line fuzzy tuning has been based on the *Controller Output Error Method* [19]. Most of the published contributions in this line present variations of the method, combined with the modifications of the membership functions [20, 21] or the addition of new membership functions [22, 23]. In the present work, the acceleration (derivative of the error) is also considered to be responsible for the controller's adaptation since, for vehicles in urban environments, the *desired speed* is supposed not to change continuously in all cases. Instead, abrupt modifications may occur due to the occurrence of unpredictable events that mean the vehicle has to make a stepwise change in speed. The evolution of the speed of the vehicle in such cases should be: (i) safe for the vehicle's occupants, guaranteeing *comfortable acceleration*, and (ii) as precise as possible.

The system was tested under stepwise changes of the desired speed of the vehicle in two different experiments: (i) over 30 different vehicles' in a simulated environment, and (ii) in a real vehicle. The simulations showed that the system is able to provide similar behaviour in different vehicles. The real environment results showed the suitability of the system for real applications, that it had remarkable precision, and was comparable with a human driver.

The rest of this communication is structured as follows. A formal statement of the problem and the initial structure of the fuzzy system that will evolve are presented in Section 2. the proposal is presented in detail in Section 3 with its division into three sub-systems. Section 4 presents the experimental simulation and real vehicle results, comparing the latter with a human driver. Finally, Section 5 presents some concluding remarks and discusses possible future lines of work.

2. Problem Statement

From a theoretical point of view, a plant to be controlled may be expressed in terms of differential equations or difference equations, provided that these are obtained from the former using a short enough sampling period [19]. The aim of a controller is to make the plant's output track a reference signal $r(k)$:

$$y(k+1) = f(y(k), ..., y(k-p), u(k), ..., u(k-q)) \quad (1)$$

where $y(k)$ is the system's output at time k, f is an unknown function, u is the control input, and p and q are constants which determine the order of the system.

In this context, the aim of many practical control problems is to produce a controller which will drive the plant's output towards a given *reference speed* representing the desired speed at which the vehicle should travel. To this end, in the present work we define a zeroth-order Takagi-Sugeno-Kang (TSK) fuzzy system with a complete AND-composed rule base defined as:

$$\begin{aligned} Rule_i : IF\ (in_1\ is\ M_1^{i_1})\ AND\ ...\ (in_N\ is\ M_N^{i_N}) \\ THEN\ out\ =\ R_i \end{aligned} \quad (2)$$

where $M_v^{i_v} \in \{M_v^1, M_v^2, ...M_v^{n_v}\}$ are the membership functions used to codify the input in_v, which has n_v different membership functions, and R_i is a numerical value representing the location of the singleton that acts as rule consequent.

The membership functions used to codify input variables are trapezoidal, defined by four real values (a, b, c, d) such that the degree of membership of an input value x is calculated as:

$$\mu(x, \{a, b, c, d\}) = \begin{cases} \frac{x-a}{b-a}, & if\ (x \in [a, b]). \\ 1, & if\ (x \in [b, c]). \\ \frac{c-x}{d-c}, & if\ (x \in [c, d]). \\ 0, & otherwise. \end{cases}$$

The t-norm *minimum* is used to implement the *AND* operator. Mamdani-type inference [24] is used, and the defuzzification operator is the weighted average. In the system, all output membership functions are singletons. Therefore, the crisp value of the output variable (out) is calculated as:

$$out = \frac{\sum R_i \cdot w_i}{\sum w_i} \quad (3)$$

where w_i represents the degree of truth of the i-th rule, and R_i is the value of the singleton inferred by the i-th rule. The weight of a rule represents its contribution to the overall control action (calculated as the minimal degree of current crisp input value membership of its respective fuzzy partitions).

Sugeno et al. [25] proved that a fuzzy system modeled with singleton consequents is a special case of a fuzzy system modeled with trapezoidal consequents, and can do almost everything the latter can. To quote from that paper: *From a theoretical point of view, we do not need a type-I controller (trapezoidal consequents) unless we want to use fuzzy terms in the consequents of fuzzy rules*, which is not our case. They also state that such a fuzzy system *is simple for identification and yet has a good approximation capability.*

Fuzzy rule based systems with singleton consequents are very commonly used in practical control system applications [26, 27, 28, 29]. In the present case, the use of singletons instead of more complex shapes to codify output variables allows fast calculation and straightforward interpretation of consequents.

Our ultimate goal with the present work is to control the speed of a vehicle in a precise way independently of its dynamics or the road conditions (slopes). Hence, given an initial fuzzy controller with all the consequents (singletons) located at zero ($R_i = 0,\ \forall i$), our immediate objectives were: (i) to learn on-line the appropriate position of the singletons, and (ii) to determine whether it is necessary to add a new membership function or to modify an existing one.

The fuzzy controller consisted of two input variables:

1. *Error*: Codify the difference between the actual speed of the controlled car and the desired speed in km/h.
2. *Acceleration*[1]: Codify the variation of the speed in km/h/s.

Both variables were codified with an initial number of trapezia (that can be modified during the process). The initial trapezia were generated by uniformly distributing their centres and displacing the top points 10 % of the size of the base, as shown in Figure 1. They overlapped to ensure that every input combination would be covered by more than one rule. Values outside the range were assumed to be equal to the corresponding limit, thereby offering maximum coverage.

[1] We considered it clearer to relate to human driving to say that the vehicle is decelerating at a rate of $-1 km/h$ per second than at $-0{,}27 m/s^2$.

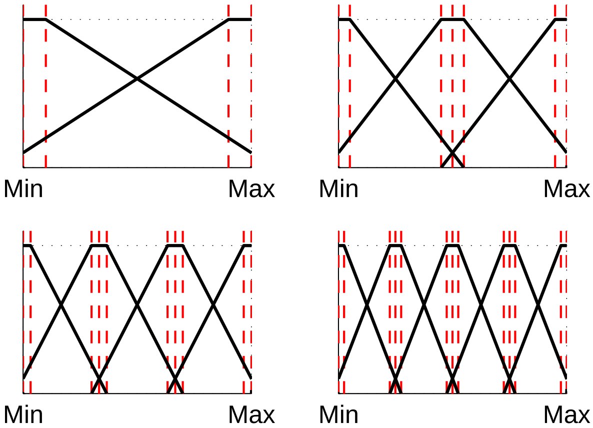

Figura 1: Distribution of the initial trapezia. Examples for 2, 3, 4, and 5 trapezia. Initial and displaced top points marked by dashed lines.

The output is codified by as many singletons as AND-composed rules exist in the rule base. The singletons are limited to the interval $[-1, 1]$. Negative values represent actions on the brake while maintaining the throttle at zero, and positive values actions on the throttle with no brake action. At the beginning of the process all the singletons are located at zero.

3. The Solution

The proposal is divided into three stages. (i) In the *singleton learning* stage, the positions of the singletons that define the output variable are adapted according to the activation of the rules involved, as well as to the current error and acceleration of the vehicle. (ii) In the *structure learning* stage, the structure of the fuzzy controller is modified by adding a new trapezium to an input variable or modifying an existing one. (iii) In the *pedal adjustment* stage, the control actions are filtered to make them more human-related. Figure 2 shows an overview of the proposed solution.

3.1. Singleton Learning

This stage adapts the consequents of the rule base, with the aim of reaching and tracking the reference more precisely. The adaptation process is based on evaluating both the error and the acceleration. It is done in this way since the desired speed signal is assumed to be stepwise up-dated in the system rather than continuously.

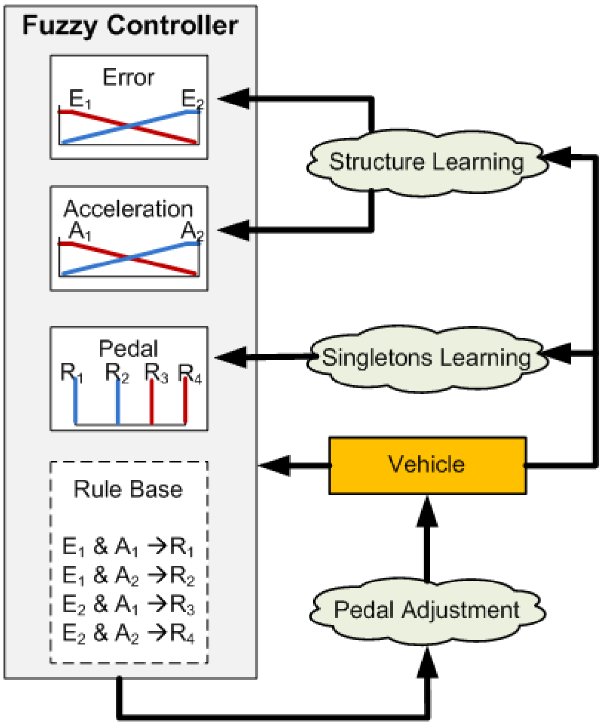

Figura 2: Schematic view of the three stages in the proposed solution. Example for a 2×2 controller.

At each instant, only the rules that were triggered are modified. Since not all the rules contributed to reaching the current state, this modification is proportional to the activation of the rules:

$$R_i(k) = R_i(k-1) + \mu_i(k-1) \cdot Reward(e(k), a(k))$$

where R_i denotes the position of a singleton, $\mu_i(k-1)$ represents the activation of the rule at previous instant, and $e(k)$ and $a(k)$ are the current error and acceleration, respectively.

The rewards direct the controller to maintaining a constant acceleration equal to some comfortable value when the error is large, and reduce the acceleration linearly down to a value of zero when the speed error reaches $e = 0$. For this purpose, the nine cases listed in Table 1 were considered:

- The set $\{C_1, C_2, C_3, C_4\}$ represents situations in which the vehicle is traveling more slowly than desired. In particular:
  - The set $\{C_1, C_2\}$ describes the situation when the vehicle is traveling very slowly with respect to the desired speed. In this case the vehicle is expected to accelerate

#	Case conditions			*Reward*
C_1	$e > 0$	$e > a_c^+$	$a > a_c^+ + T$	$-C \cdot \|e\|$
C_2			$a < a_c^+ - T$	$C \cdot \|e\|$
C_3		$e \leq a_c^+$	$a > e + T$	$-C \cdot \|e\|$
C_4			$a < max(0, e - T)$	$C \cdot \|e\|$
C_5	$e < 0$	$e < a_c^-$	$a < a_c^- - T$	$C \cdot \|e\|$
C_6			$a > a_c^- + T$	$-C \cdot \|e\|$
C_7		$e \geq a_c^-$	$a < e - T$	$C \cdot \|e\|$
C_8			$a > min(0, e + T)$	$-C \cdot \|e\|$
C_9	**Otherwise**			0

Cuadro 1: Cases to consider in implementing the singleton learning.

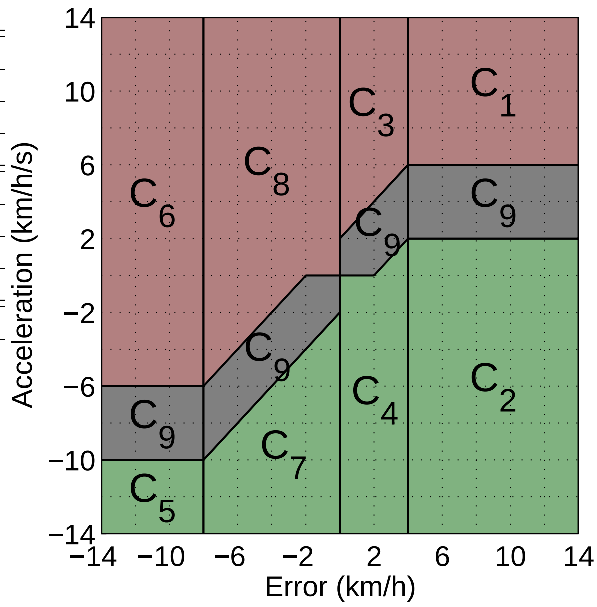

Figura 3: Cases covered by the singleton learning. Red area: zone where singletons are reduced; green area: zone where singletons are augmented; gray area: zone where singletons are unmodified.

with positive constant acceleration equal to the comfort value (a_c^+):

- C_1: the acceleration is greater than the comfort value plus a threshold, so singletons must be reduced.
- C_2: the acceleration is less than the comfort value minus a threshold, so singletons must be augmented.

- $\{C_3, C_4\}$ describes the situation when the vehicle is traveling slowly but near the desired speed. In this case, the vehicle is expected to reduce the acceleration linearly until reaching the reference speed:
  - C_3: the acceleration is greater than the error plus a threshold, so singletons must be reduced.
  - C_4: the acceleration is less than the error minus a threshold, so singletons must be augmented.

- The set $\{C_5, C_6, C_7, C_8\}$ represents situations where the vehicle is traveling faster than desired. The cases are described and rewards are applied mirroring those for $\{C_1, C_2, C_3, C_4\}$, but considering a different constant negative comfort acceleration (a_c^-).

- Finally, C_9 represents the case when no change must be applied to the singletons since the speed and acceleration of the vehicle are within the desired range.

The cases are dependent on the following parameters. First, a_c^+ represents the comfort acceleration when the vehicle is increasing in speed, i.e., the desired maximum acceleration when the vehicle's speed is far from the reference value. The value used for the experiments was fixed at $4km/h/s$. Second, a_c^- represents the comfort acceleration when the vehicle is braking. In this case, $-8km/h/s$ was set for the experiments. Third, T represents a threshold used to mitigate the possible effect of noise in the measurements. We set $T = 2km/h/s$ for experiments. And fourth, $C = 0{,}01$ is used as a normalization constant.

With this configuration of the parameters, the cases used in the learning of the singletons define the zones shown in Figure 3, where the red and green zones indicate cases when rewards are negative ($\{C_1, C_3, C_6, C_8\}$) or positive ($\{C_2, C_4, C_5, C_7\}$), respectively, and the gray zone represents the desired situation where no reward is applied to the singletons (Case C_9).

3.2. Structure Learning

This stage evaluates the behaviour of the current controller during a certain amount of time ($cycle = 100$, in seconds), and decides whether it is necessary (i) to add a new trapezium, or (ii) to modify an existing one.

To decide which, if any, modification is applied, first the histogram of the input values is generated,

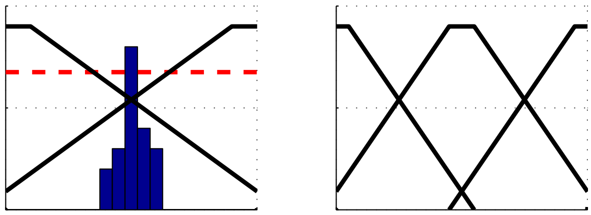

Figura 4: Example of label addition. Original labels with superimposed histogram (left); resulting labels (right).

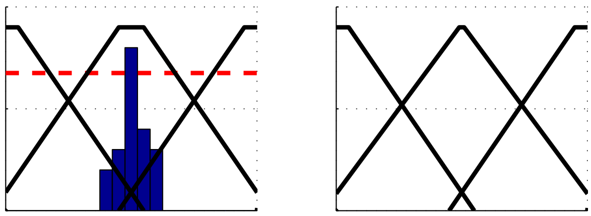

Figura 5: Examole of label modification. Original labels with superimposed histogram (left); resulting labels (right).

and then an analysis is made of how the commonest values are covered by the current trapezia. This process is carried out as follows:

- If the most repeated value in the histogram is covered with an activation degree less than 0.75 then a new membership function is inserted into the variable. The trapezia are reinitialized (Figure 1), and singletons are reset to zero. This process is illustrated in Figure 4.
- If both of the two most repeated values are covered with an activation degree greater than 0.75 then the shorter base of the trapezium is reduced by 80 %. Singletons are not reset after this. This process is illustrated in Figure 5.

Adding a new trapezium is designed to cover the most repeated input range to a greater degree so as to generate a clear control action, while reducing the shorter base is aimed at obtaining a controller that is more specific leading to better differentiation in the commonest input range.

3.3. Pedal Adjustments

Three aspects are taken into account to provide a more human-like control of the pedals:

1. When the sign of the control signal changes, the system returns zero for 0.5 seconds in order to simulate the delay of *the foot changing from one pedal to the other.*
2. When the reference speed changes, the singleton learning process is deactivated for 1 second, to allow the controller to act without any disturbance produced by possible modifications of the singletons.
3. The pedal is set to zero when its absolute value is less than 0.02, since at this low level it has no real effect.

With these modifications, the system is expected to emulate the actions of a human driver more precisely, as well as to smooth out any potential abrupt modifications of the singletons produced by large changes in the reference speed.

4. Experimentation and Results

Experiments were carried out in two phases: (i) in a simulated environment in order to analyse the system without risk and for a broad set of vehicle dynamics; and (ii) in a real vehicle both to study the performance in real driving situations and to compare it to a human driver.

4.1. Tests in the Simulated Environment

For the experiments in a simulated environment, TORCS[2] (*The Open Racing Car Simulator*) was used as testbed. This is one of the most popular car racing simulators for academic research due to its various advantages: (i) it lies between an advanced simulator and a fully customizable environment, such as those used by computational intelligence researchers; (ii) it features a sophisticated physics engine; and (iii) it implements an ample set of tracks and vehicles with different physical behaviour.

There are 30 models of vehicles implemented in TORCS, all of them differing in their longitudinal behaviour. To illustrate this, Figure 6 gives the values for some of the parameters that affect the longitudinal dynamics of all TORC's vehicles. In this figure, the red lines and the blue boxes represent the values of the mean and standard deviation. The values are normalized with respect to the minimum and maximum values found.

All the vehicle models were used in the experiments in order to test the robustness of the control system for different dynamics. The experiments consisted of giving the vehicles the following reference speeds: $\{20, 35, 30, 20, 40\} km/h$, for 20 seconds each, and repeated 8 times.

[2] http://torcs.sourceforge.net/

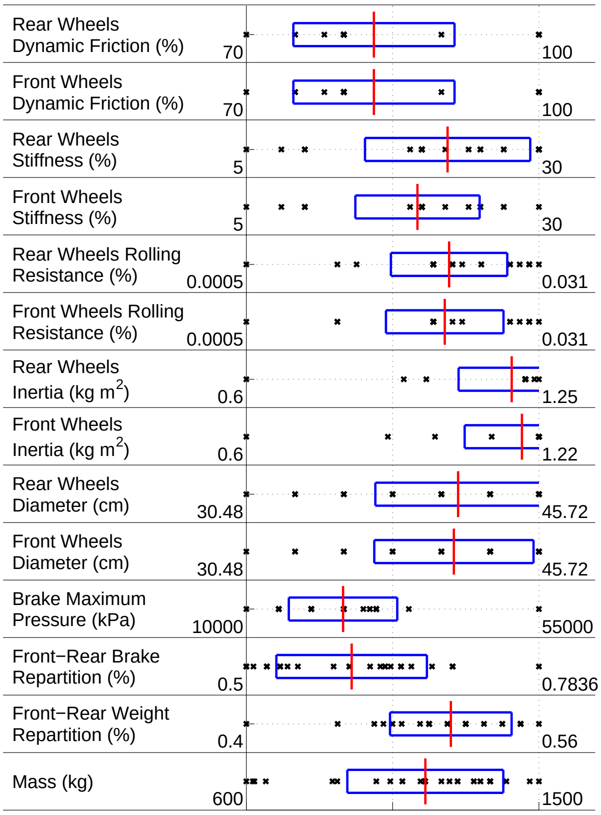

Figura 6: Comparison between some of the longitudinal attributes of the vehicles in TORCS. Red line: mean value; blue box: mean ± standard deviation.

The track was an oval comprising two straights of 1.6 km joined by semi-circles, with the aim of not conditioning the system's behaviour to managing the steering. Since the gear must also be controlled, a simple policy was implemented which shifts up the current gear if the revolutions per minute (rpm) of the vehicle's engine are over 4000, and shifts down when $rpm < 2500$. The parameters of the learning system were set as follows:

- Ranges of $[-25, 25]km/h$ for the *Error* and $[-8, 8]km/h/s$ for the *Acceleration*.
- The controller started with 2 trapezia per input (4 rules).

Figure 7 shows the speed results of the 30 vehicles superimposed in the top graph, and in the bottom, zoomed zones of the graph with only the fastest, the slowest, and the averaged speeds shown. The results seem to reflect good precision: one observes in the zoomed plots the effect of learning, since

Figura 8: Vehicle (top), and test zone with path to follow (bottom).

the difference between the highest and the lowest speeds decreases over time (until t he maximum error $\leq 1km/h$).

Given the promising results in the simulated environment, we proceeded to test the system in a real vehicle, as will be described in the next subsection.

4.2. Tests in the Real Environment

A Citroën C3 (Figure 8, top) modified to permit autonomous control of the pedals, was used for these trials [30]. The gear is unknown to the controller since the control implemented by Citroën was used. In particular, there was no knowledge about the current gear, or how or when it changed. Figure 8 (bottom) shows an aerial view of the path to follow over the test zone. It has slopes of up to 3 %, and a long straight segment of about 200 metres. The points marked are references for experimenting with variable speeds.

Some modifications were made to the configuration used in the simulated environment:

- The input ranges were reduced to $[-20, 20]km/h$ for the *Error* and to $[-5, 5]km/h/s$ for the *Acceleration*.

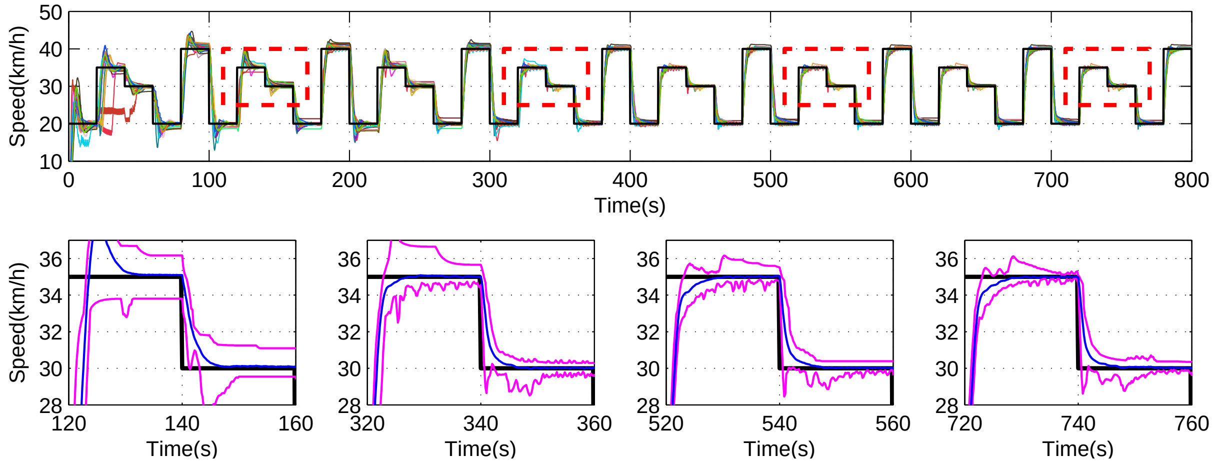

Figura 7: Execution of the learning process in 30 vehicles (top). Zooms showing the highest, lowest, and averaged speeds (bottom).

- The starting controller codified *Error* with 4 trapezia instead of the 2 used in the simulation experiments, thereby obtaining an initial controller with 8 rules.

- Finally, singletons were restricted to $[-0{,}3, 0{,}5]$ since values outside that range could cause damage to the vehicle's equipment.

At first, the system was tested using two constant reference speeds – 15 and 5 km/h. The results are shown in Figures 9 and 10 in which the speed, pedal action, and evolution of the consequents over time are shown. The speed results are compared with the behaviour of a human driver, who was helped by being shown on a screen the vehicle's real speed, since the speedometer was insufficiently accurate for adequate control.

In both tests, the structure learning was executed at $t = 100s$, converting a 4×2 controller into a 5×3 one, so that resetting the singletons produced the speed reduction. Furthermore, at $t = 200s$ the central labels of both variables were stretched without any significant effect. At the top of each figure, two *Mean Absolute Error* (MAE) values are shown. The one after $t = 25s$ (transitory state) and the overall value. It is important to remark that most of the singletons seem to reach a state of stability once the granulation of the controller has been modified. In both experiments, only one singleton significantly varied over time, and in both cases corresponding to the rule that covered both the error and the acceleration equal to zero. The oscillations

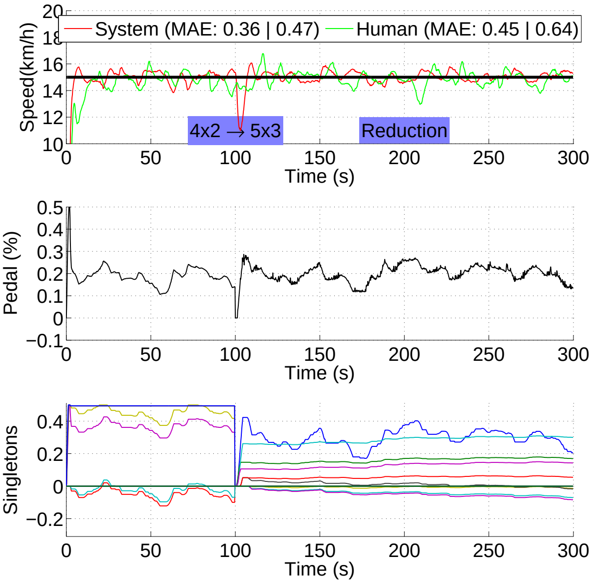

Figura 9: Results maintaining a fixed reference speed of 15km/h. Evolution of the speed (top), pedal action (centre), and singletons (bottom).

of this singleton occur to adapt the system to the variations in the road or the vehicle's dynamics.

In both cases, the speed management provided by the learning system outperformed that of the human driver. It is important to remark that 15 km/h was selected because it represented a *frontier* between first and second gear in the case that the vehicle is accelerating rapidly. During the test, the vehicle maintained first gear, indicative of the

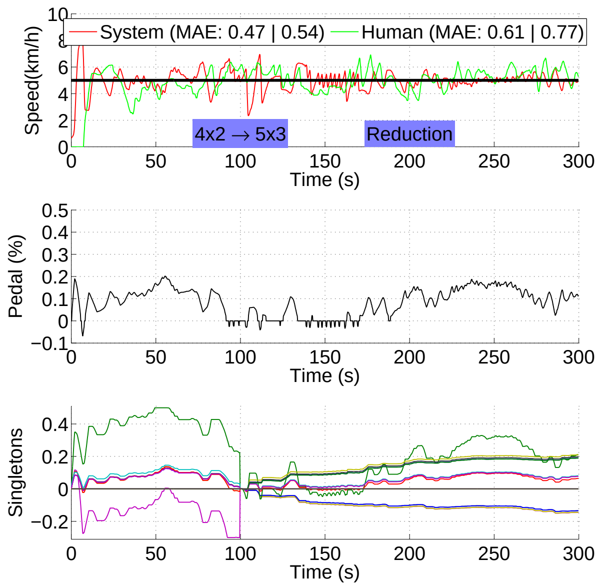

Figura 10: Results maintaining a fixed reference speed of 5km/h. Evolution of the speed (top), pedal action (centre), and singletons (bottom).

quality of the acceleration given to the vehicle. The speed of 5 km/h was an interesting challenge since at this speed the slightest slope or variation on the pedal can induce major changes in speed. The controller maintained $MAE \approx 0{,}5km/h$, which not only reflects good accuracy but is also insignificant for the vehicle.

A second experiment was conducted in which the reference speed was changed over time. The vehicle started at Point A (Figure 8), and the reference speed was changed at each marked point. The evolution of the speed is shown in Figure 11. During the experiment, at $t = 100s$ the $4x2$ controller was converted into a $5x3$ one, which is the reason for the poor behaviour around that instant. Also, at $t = 200s$, the central label of the $Error$ was reduced.

For a quantitative analysis of the behaviour during the experiment, Figure 12 shows measures of the precision of the execution during the experiment, distinguishing the accelerating (top) and braking (bottom) steps. In this figure, the values were calculated with respect to the desired speed of the vehicle assuming it is following the indications of the singleton learning module (Figure 3). As can be seen, MAE is smaller in the braking steps. This is because the dynamics of the vehicle when using the brake are faster than when using the throttle, so that it is easier to follow the acceleration indications. In all the steps, the stationary MAE evolves until $MAE \approx 0{,}5km/h$, and the transitory value until $MAE \approx 1{,}0km/h$. In the overall execution, the average MAE decreases over time. The exception ($25 \Downarrow 15$) is due to the resetting of the singletons made by the structure learning module at $t = 100s$.

5. Conclusions and Future Work

This communication has presented a method for the on-line evolution of a fuzzy controller responsible for managing the pedals of a vehicle, based on data obtained while the vehicle is moving. The method is divided into three phases: (i) a singleton learning phase, responsible for modifying the positions of the singletons of the controller depending on the speed and acceleration of the vehicle; (ii) a structure learning phase that, after a certain amount of time, varies the number or shape of the trapezia used to codify the input variables; and (iii) a pedal adjustment phase in which the actions given by the controller are filtered to make them more reliable.

The system was tested in both a simulated environment, and on a real vehicle. In the simulations, it was tested on 30 cars with different dynamical behaviour, and yielded accurate results with low deviations over time for all the cars. In the real vehicle trials, the results were compared with those of a human driver. The control system outperformed the human under conditions of constant reference speeds, and gave excellent results in both speed and acceleration for a changing reference speeds.

Future work will focus on greater sophistication in the structural learning, since the present implementation resets the singletons after a granularity change. This can be resolved by interpolating the new rule base with respect to the previous one. In the same line, the present structural learning changes both the number and amplitude of the trapezia, but not their centres. It is planned to use data concerning the input histogram to redistribute the new trapezia accordingly.

New transport applications are expected to be implemented with the proposed method. An ACC system able to maintain a safe distance with a preceding vehicle can be easily implemented by using the difference with the desired distance as error signal, and then applying the same approach as has

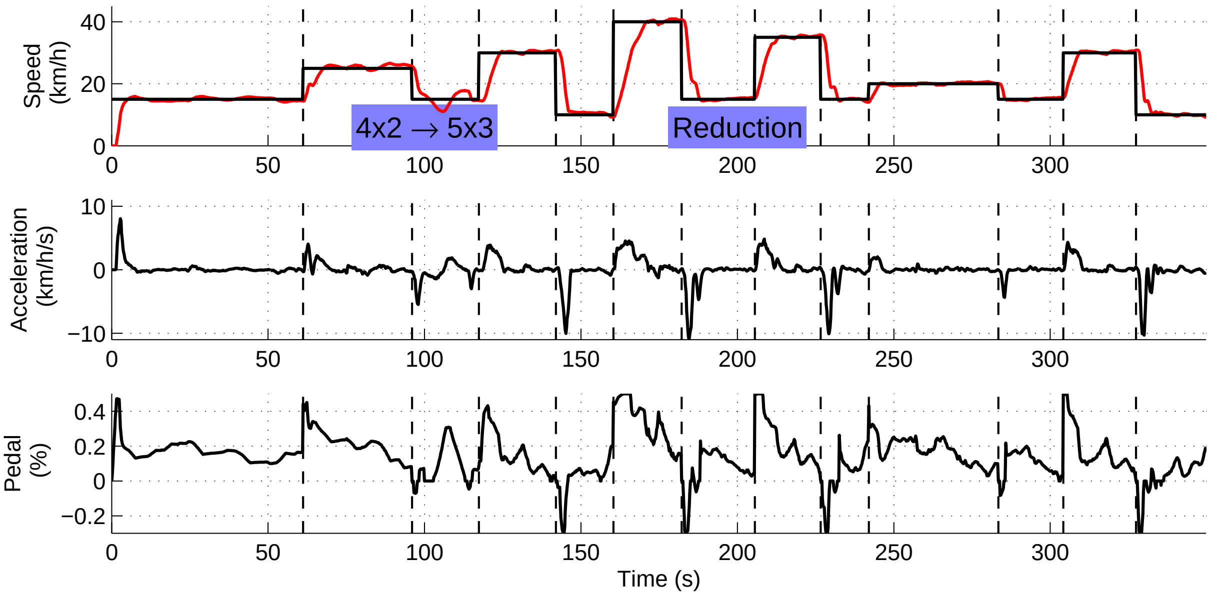

Figura 11: Speed of the vehicle with changing reference speed.

been presented in this work. Similarly, steering control can be based on using the error with respect to the reference path to follow.

ACKNOWLEDGMENT

Onieva thanks the projects TIN2008-06872-C04-04 and TIN2011-27696-C02-01 from the Spanish Ministry of Economy and Competitiveness, P07-TIC-02970 and P11-TIC-8001 from the Andalusian Government. Godoy thanks the to the JAE program (CSIC). Milanés wants to express his gratitude to the ME/Fulbright Program. Godoy and Villagrá thanks to the projects TRA2011-14112-E and TRA2008Ũ06602-C03Ũ01 from the Spanish Ministry of Economy and Competitiveness.

[1] W. D. Jones, “Keeping cars from crashing,” *IEEE Spectrum*, vol. 38, no. 9, pp. 40–45, 2001.

[2] V. Milanés, J. Pérez, J. Godoy, and E. Onieva, “A fuzzy aid rear-end collision warning/avoidance system,” *Expert Systems with Applications*, vol. 39, no. 10, pp. 9097–9107, 2012.

[3] J. Pérez, V. Milanés, and E. Onieva, “Cascade architecture for lateral control in autonomous vehicles,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 1, pp. 73–82, 2011.

[4] E. Onieva, V. Milanés, C. Gonzlez, T. De Pedro, J. Pérez, and J. Alonso, “Throttle and brake pedals automation for populated areas,” *Robotica*, vol. 28, no. 4, pp. 509–516, 2010.

[5] Eurobarameter, “Use of intelligent systems in vehicles,” *European Commission*, 2006.

[6] J. E. Naranjo, C. Gonzalez, J. Reviejo, R. Garcia, and T. de Pedro, “Adaptive fuzzy control for inter-vehicle gap keeping,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 4, no. 3, pp. 132–142, 2003.

[7] C. Desjardins and B. Chaib-Draa, “Cooperative adaptive cruise control: A reinforcement learning approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1248–1260, 2011.

[8] Y.-F. Peng, “Adaptive intelligent backstepping longitudinal control of vehicleplatoons using output recurrent cerebellar model articulation controller,” *Expert Systems with Applications*, vol. 37, no. 3, pp. 2016 – 2027, 2010.

[9] J.-J. Martinez and C. Canudas-de Wit, “A safe longitudinal control for adaptive cruise control and stop-and-go scenarios,” *IEEE Transactions on Control Systems Technology*, vol. 15, no. 2, pp. 246–258, 2007.

[10] A. T. Chronopoulos and C. M. Johnston, “A real-time traffic simulation using a communication latency hiding parallelization,” *IEEE Transactions on Vehicular Technology*, vol. 51, no. 3, pp. 498–510, May 2002.

[11] ——, “A real-time traffic simulation system,” *IEEE Transactions on Vehicular Technology*, vol. 47, no. 1, pp. 321–331, Feb. 1998.

[12] M. C. Marques and R. Neves-Silva, “Traffic simulation for intelligent transportation systems development,” in *Proc. IEEE Intelligent Transportation Systems*, 2005, pp. 320–325.

[13] A. Gallione, V. Murdocco, and S. Campo, “A comparative study for autonomous highway driving system with satellite-based positioning,” in *Proc. IEEE Conference on Control Applications CCA 2003*, vol. 1, 23–25 June 2003, pp. 1–6.

[14] M. T. Wolf and J. W. Burdick, “Artificial potential functions for highway driving with collision avoidance,” in *Proc. IEEE International Conference on Robotics and Automation*, 2008, pp. 3731–3736.

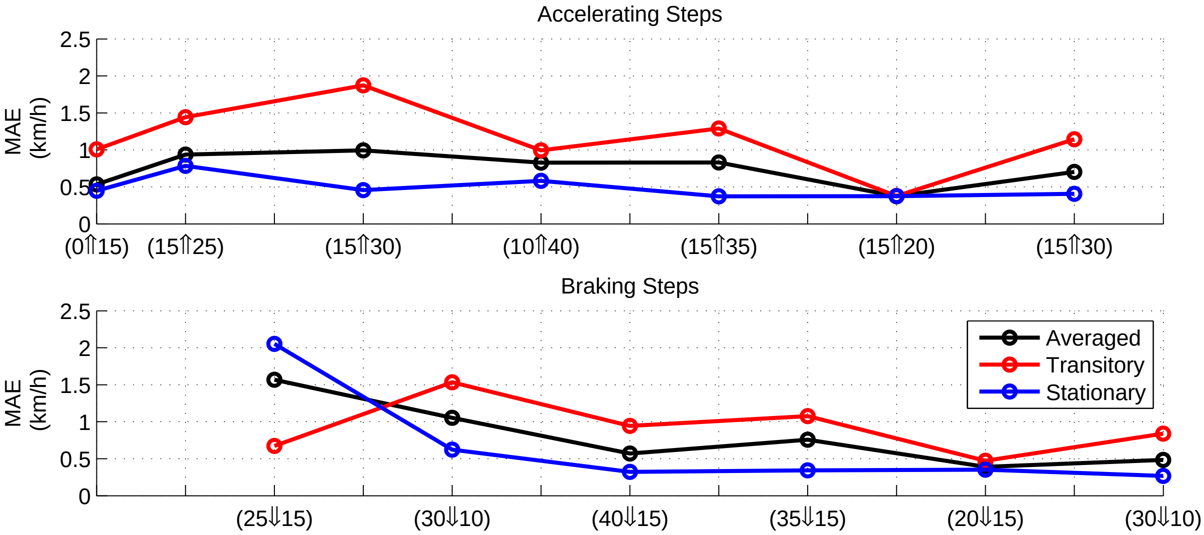

Figura 12: Analysis of the step test MAE: averaged during the test (black), and during the transitory (red) and stationary (blue) states. Accelerating steps (top), and braking steps (bottom).

[15] L. C. Davis, "Effect of adaptive cruise control systems on traffic flow," *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, vol. 69(6), p. 066110, 2004.

[16] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[17] A. Rajapakse, K. Furuta, and S. Kondo, "Evolutionary learning of fuzzy logic controllers and their adaptation through perpetual evolution," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 3, pp. 309–321, 2002.

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

[19] H. Andersen, A. Lotfi, and A. Tsoi, "A new approach to adaptive fuzzy control: the controller output error method," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 27, no. 4, pp. 686–691, aug 1997.

[20] H. Pomares, "Online global learning in direct fuzzy controllers," *IEEE Transactions on Fuzzy Systems*, vol. 12, no. 2, pp. 218–229, 2004.

[21] G. M. E. I. Y. E. Karasakal, O., "An error-based on-line rule weight adjustment method for fuzzy pid controllers," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10 124–10 132, 2011.

[22] A. Cara, H. Pomares, and I. Rojas, "A new methodology for the online adaptation of fuzzy self-structuring controllers," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 3, pp. 449–464, 2011.

[23] H. Zhuang and X. Wu, "Membership function modification of fuzzy logic controllers with histogram equalization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 31, no. 1, pp. 125–132, 2001.

[24] E. H. Mamdani, "Applications of fuzzy algorithms for simple dynamic plant," *Proceedings IEEE*, vol. 62(12), pp. 1585–1588, 1974.

[25] M. Sugeno, "On stability of fuzzy systems expressed by fuzzy rules with singleton consequents," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 2, pp. 201–224, April 1999.

[26] C. F. Juang, C. T. Chiou, and C.-L. Lai, "Hierarchical singleton-type recurrent neural fuzzy networks for noisy speech recognition," *IEEE Transactions on Neural Networks*, vol. 18, no. 3, pp. 833–843, 2007.

[27] E. Jahanshahi, K. Salahshoor, and Y. Sahraie, "Application of fuzzy observer and controller in gas-lifted oil wells," in *Proc. IEEE International Conference on Networking, Sensing and Control*, 2008, pp. 101–106.

[28] L. L. Simon and K. Hungerbuehler, "Real time takagi-sugeno fuzzy model based pattern recognition in the batch chemical industry," in *IEEE International Conference on Fuzzy Systems*, 2008, pp. 779–782.

[29] S. Jinju, W. Minxiang, and W. Weidong, "Robust takagi-sugeno fuzzy control for a mini aviation engine," in *27th Chinese Control Conference*, 2008, pp. 775–780.

[30] V. Milanés, C. González, J. Naranjo, E. Onieva, and T. De Pedro, "Electro-hydraulic braking system for autonomous vehicles," *International Journal of Automotive Technology*, vol. 11, pp. 89–95, 2010.