22nd ITS World Congress, Bordeaux, France, 5–9 October 2015

Paper number ITS-1763

Behavior personalized Adaptive Cruise Control using Probability-Weighted ARX model

Thomas WILHELEM, Hiroyuki OKUDA, Blaine LEVEDAHL
Tatsuya SUZUKI, Tetsunori HARAGUCHI
Nagoya University, Japan

Abstract- Adaptive cruise control (ACC) enables automatic speed regulation of a driven vehicle based on the observation of surrounding vehicles on the road. This paper proposes a new ACC structure able to transform machine learning based car following models into ACC models. The concept is to separate driving situations into two categories: following the lead vehicle, or following a virtual vehicle if no any lead vehicle is present. Based on that and on switching cases, the use of machine learning car following models can be done, and this leads to smarter and more personalized ACC. In this paper, the detailed model structure is explained, and examples of simulated functioning are exposed with two different car following models. Then, the created ACC is compared to existing models, and an integration scheme is proposed.

Keywords- Active cruise control; machine learning; Probability-Weighted ARX

I. Introduction

Adaptive cruise control (ACC) systems have been studied and implemented on cars in recent years, and they are part of the first steps to vehicle automation [1]. As an extension of classic cruise control, most of these systems have been designed to keep a minimum predefined headway time to the leading vehicle. This functionality can reduce fatigue of the driver, fuel consumption, and traffic flow congestion [2].
In the past years, ACC models and vehicle maneuver planning systems have been created taking into account the limited resources of fail-safe vehicles electronic architectures [3]. In these cases, the ACC assistance is based on a simple risk evaluation index such as headway time, and the user has to manually predefine the value. This type of system is designed considering the average behavior among users, and it is cannot be entirely personalized.
Taking advantage of classic low computing power ACCs and behavior learning, researchers created adaptive cruise control systems able to calculate and use the correct headway time as a reference for the control program [2].
The view point taken in this paper is different. Recent evolutions of embedded systems enables us to implement a control method with more complexity and higher computational load. Moreover, due to the increase of passenger vehicles connectivity, heavy computational tasks can be performed in the cloud [4]. Thus machine learning can considered as a serious candidate for online vehicle control.
In recent years, driving behavior has been approached through machine learning and investigated by numerous researches. The main idea is to observe the human as a “controller”. It can be done by using linear or non-linear control theory [5], stochastic models, hybrid models, or implying neural networks or hidden Markov chains [6-8].
Based on this knowledge, this paper proposes an ACC system able to use machine learning based following systems. Emphasis is done on the use of the Probability-Weighted ARX model [9]. This paper demonstrates the functioning of the ACC system, its strong and weak points, and compares it to classic ACC as well as following models.

Section 2 presents the concept of the novel ACC system and its basic operating. Section 3 details architecture of the model. Sections 4 and 5 explain in detail the used vehicle following models: the machine learning PrARX model, and the GHR car following model. Section 6 demonstrates the use of the created ACC structure. Section 7 compares the results with classic ACC and traffic flow model. Section 8 proposes a possible model use, and finally, in section 9, conclusions are drawn.

II. Virtual vehicle ACC concept

In this first section, the concept, the macroscopic architecture, and the main working situations of the presented adaptive cruise control are exposed.

a) Model concept

The concept behind this model is to create a simple structure able transform machine learning vehicle following models in ACC models. The car following model used in this study is the PrARX model [9], which is a multiple ARX model with soft mode transitions abilities. This vehicle following model, once correctly calibrated, can reproduce accurately the driver dynamics, following distance and response time. Nevertheless this ARX model needs to follow a lead vehicle at any time to be able to deliver an output, and this can be done thanks to the here exposed Virtual Leading Vehicle ACC system (Vlv-ACC).

As smart cruise control systems, ACCs have to be able to deal with two main situations. Cruising at constant velocity when there is no leading vehicle in the way, or follow a leading vehicle with the right safety distance. Moreover an ACC must be able to handle correctly transition phases.
To be able to tackle these issues, the concept of virtual leading vehicle (VLV) was created. The virtual leading vehicle replaces the real leading vehicle (RLV) as soon as the real leading vehicle following conditions are not satisfied. The VLV is driven at the cruise control desired velocity, and a set of VLV/RLV switching rules enable to change the leading vehicle smoothly. Thus the following vehicle (FV) driven by a standard car following model gets the ACC functionality.

Figure 1: Two lanes highway driving. The FV is following the VLV.

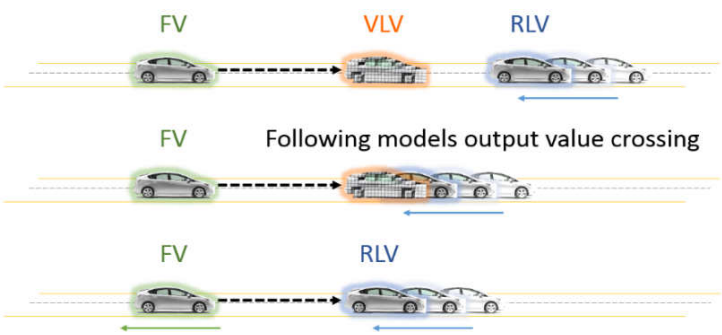
The Vlv-ACC uses the selected following model to follow both the VLV and RLV at any time. Then based on the vehicles relative positions and on the VLV and RLV following models outputs, the switching mechanism can select which target vehicle the FV has to follow. The following rules used to switch between situations only works with sufficiently sophisticated following models. Indeed the selected following model must have a natural reaction to the distance and relative velocity between vehicles.

b) Model situations definition

In this configuration, two main situations and four switching cases can occur. The two main situations have already been exposed: following the virtual leading vehicle (VLV) or following the real leading vehicle (RLV). The VLV following situation is the basic state when the ACC is activated. The VLV is at first generated at the point of non-acceleration of the following model.

Then the four possible switching situations are the following:

1) <u>Soft RLV in</u>



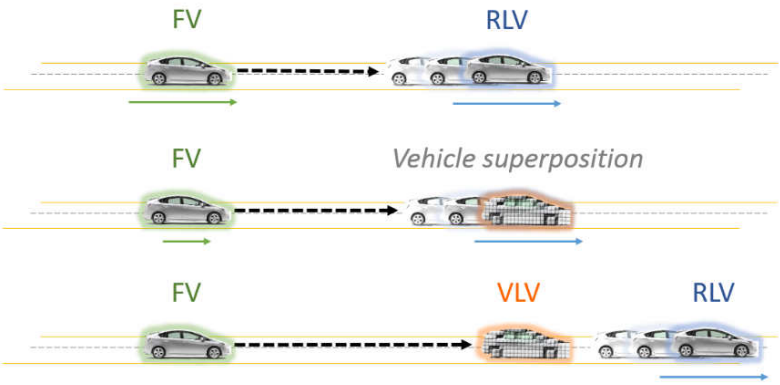
Following models outputs (acceleration)	
VLV	RLV
0	Positive
0	0
- - -	Negative

Figure 2: "Soft RLV in" situation. The RLV is slowing down, and is smoothly replacing the VLV, when the model's outputs (FV acceleration) values intersect.

This switching case modifies the situation from following the VLV to following the RLV.
"Soft RLV in" expresses the fact that while following the VLV, a slower than VLV and far away RLV vehicle is in the current lane. After a certain time, this RLV will overlap the VLV and eventually collide the FV.
To avoid this situation, both following models output for the RLV and VLV are constantly computed, and when these models outputs (in our case vehicle acceleration) intersect, the transition from following VLV to RLV is done.

2) Soft RLV out

This switching case modifies the situation from following the RLV to following the VLV.
"Soft RLV out" expresses the fact that while following a RLV, the RLV accelerates to a velocity higher than the ACC desired velocity. When the RLV reaches the cruise control desired velocity, the RLV is replaced by an overlapping VLV. Another switching solution is to virtually superpose the RLV and VLV, but force the VLV velocity to the ACC velocity for the following model output calculation. Then the transition can be triggered using the same technique as "Soft RLV in", with following models output values crossing.

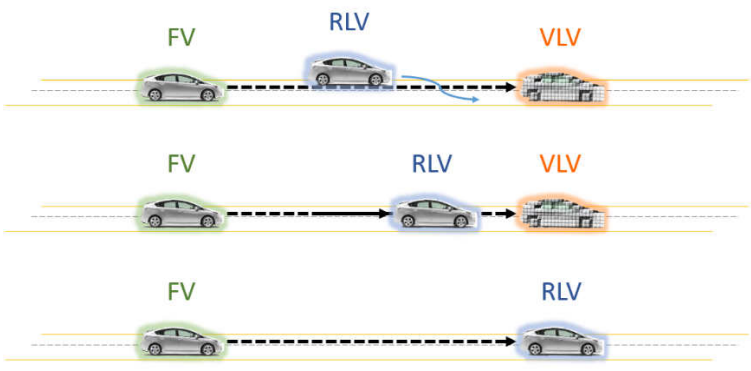


Leading vehicle velocity	
VLV	RLV
- - -	Lower than ACC defined
ACC defined velocity	ACC defined velocity
ACC defined velocity	Higher than ACC defined

Figure 3: "Soft RLV out" situation. Method 1. FV is following RLV. RLV accelerates. When RLV velocity is higher that the cruise control desired velocity, the RLV is replaced by the VLV.

3) Hard RLV in

This switching case modifies the situation from following the VLV to following the RLV (also named real leading vehicle cut-in situation).
"Hard RLV in" expresses the fact that a RLV inserts between the FV and the VLV, and its velocity is low enough so the driver would need to brake. In this case, output of the following models are switched in a time corresponding to a fraction of the initial headway time, avoiding any safety related issue as well as acceleration discontinuity. The output switch occurs only if the replacing following model output value is lower than the previous one.

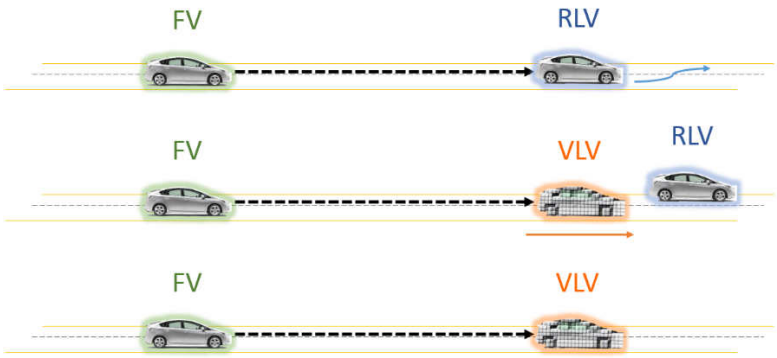


Following models outputs			
T	VLV	RLV	FV
$T < T_0$	a_x	- - -	a_x
$T_0 < T < T_0 + \tau$	$a_x(T_0)$	a_y	$t*a_x(T_0) + (t-\tau)*a_y$
$T_0 + \tau < T$	- - -	a_y	a_y

Figure 4: Example of a possible "Hard RLV in" situation. Here the previously following vehicle was the VLV and a RLV inserts with a low differential velocity. a_y has to be lower than a_x to trigger the "Hard RLV in situation".

4) Hard RLV out

This switching case modifies the situation from following the RLV to following the VLV.
"Hard RLV out" expresses the fact that while following the RLV, this vehicle quickly exits from the current lane (lane change, highway exit …). The RLV is replaced by the VLV. The VLV velocity is increased from RLV velocity to the cruise control desired velocity with an acceleration corresponding to the average observed driver acceleration during the learning phase.



Vehicles accelerations		
Conditions	VLV	RLV
---	---	a_{RLV}
$V_{VLV} < V_{ACC}$	$a_{defined}$	---
$V_{VLV} = V_{ACC}$	0	---

Figure 5: Example of 'Hard RLV out' situation.

The exposed switching conditions are only one possible interpretation of the model. These conditions have shown good results with machine learning vehicle following models, but can be too broad for some basic car following models.

III. Vlv-ACC implementation

This section is dedicated to the understanding of the model structure from the point of view of implementation. Simulink has been used to build the model. It enables easy replacement of the vehicle following model, and a connection can be done with the vehicle simulation software IPG Carmaker and AVL Cruise.

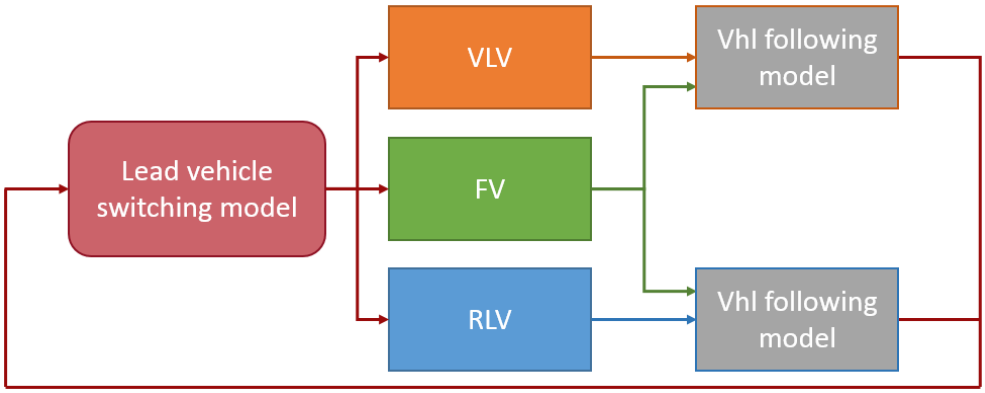


Figure 6: Macroscopic model diagram.

As explained in section I, the Vlv-ACC runs two instances of the selected vehicle following model in parallel. This is required to be able to have smooth mode switching in case of "soft RLV in". This transition condition is based on both following models output comparison. In Fig. 6, the upper and lower vehicle following branch use the same following model. Only the inputs are different.
The upper branch is dedicated to the VLV. The VLV bloc is able to generate the virtual leading vehicle based on the desired cruise control velocity, the initial relative distance to the following vehicle, or the actual position of the RLV. The initial relative distance is optimized a priori to enable no acceleration at the vehicle following model output.
The lower branch is dedicated to the RLV. In a real world situation, the RLV information would be extracted from on vehicle sensors such as a Radar, Lidar or camera system, and the flow vehicle

information could be communicated to the following model (see Section VIII.b)). However, in the case of this simulation, the RLV position is read from a vehicle position file. IPG Carmaker with on vehicle sensors and traffic flow would also be a solution.
The following figure shows the Simulink implementation of the following model.

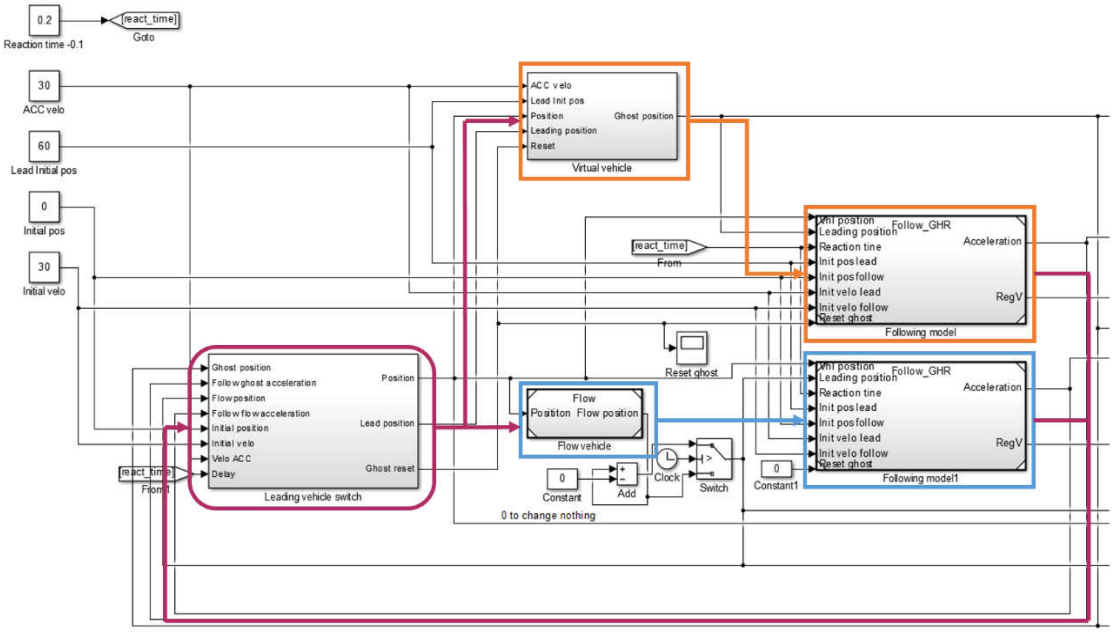


Figure 7: Example of Simulink integration of the model.

IV. PrARX car following model

The machine learning car following model used to demonstrate the ability of the ACC system is the PrARX model [9]. This model enables the reproduction of the Vehicle/Driver entity dynamics. Thus, a complex model of the vehicle dynamical response is not necessary. The idea behind PrARX is similar to Piecewise affine autoregressive exogenous (PWARX) models, which is an expression of hybrid dynamical systems. Whereas PWARX models have discrete mode switching, PrARX introduces soft mode switch based on probability estimation. This avoids discontinuity in the model output during mode transition. PrARX models also allow parameter estimation in an adaptive manner, which opens the possibility to a progressive revision of the following parameters depending on the user's behavior modification. Nevertheless, the level of sophistication of this model also leads to more difficult parameter identification and the model stability analysis is quite delicate.

a) Definition of the model

The PrARX model is composed of ARX models weighted by parameters expressing the probability of the presence in a mode. Thus the PrARX model can be defined by:

$$f_{PrARX}(r_k) = \sum_{i=1}^{s} P_i \theta_i^T \varphi_k \tag{1}$$

where $r_k \in \mathbb{R}^n$ defines the regressor vector (input vector), $\varphi_k = [r_k^T \; 1]^T \in \mathbb{R}^{n+1}$, $\theta_i^T \in \mathbb{R}^{(n+1)*q}$ (i $= 1, \dots,$ s)is the identified vectors defining the i^{th} ARX modes, s defines the number of modes, and $P_i \in \mathbb{R}^q$ is the vector expressing the mode probability of the q outputs. P_i is defined by the following softmax function:

$$P_i = \frac{\exp(\eta_i^T \varphi_k)}{\sum_{j=1}^{s} \exp(\eta_j^T \varphi_k)} \qquad \eta_s = 0 \tag{2}$$

where η_i $(i = 1, \dots, s-1)$ is the identified parameter defining the probabilistic partition of the different modes.
More details about the PrARX model are available in [9].

b) Drivers identification

Driver measurements have been done in a driving simulator. The driving Simulator is composed of a real vehicle interior, and the visualized image is created with 3 wide video-projected screens. This configuration enables 180 degree vision, and gives good driver immersion. Vehicle dynamics is calculated by the use of Carsim (Virtual Mechanics Corporation), and the environment represents a typical a four lanes highway. The implemented velocity patterns represent high and low congestion situations on this road.

Reproduction of the driving behavior depends not only on the structure of the model (here PrARX), but also on the selection of explanatory variables of the model. The selected input vector is composed of acceleration of the driven vehicle, distance between the driver and leading vehicles (range), relative velocity between the vehicles (range-rate), and the inverse of the headway time (velocity/range) [10].

This set of inputs enables to give vehicles dynamic information and surrounding vehicles relation characteristics.

The output of the model is the desired acceleration of the vehicle.

A delay of 300 ms is applied between the input and the output of the model, to insure a cognition time matching the average human brain capacity [11].

The driven parameters identification process is done by optimizing independently the ARX modes parameters based on pre-clustered data. Supervised clustering of the learning data enables robustness in the parameter identification, and the mode separation parameters are calculated uses multinomial logistic regression.

The selected mode data clustering is:

A mode for the 20% most negative accelerations, a mode for the 15% highest accelerations, and the last mode for the rest of the data. This combination gave a stable and safe reproduction of the vehicle/driver dynamics.

V. Gazis-Herman-Rothery car following model

The Vlv-ACC is a flexible platform able to receive different types of car following models. While some very basic following models can need some adjustments on the ACC switching conditions, most of the classic car-following models work without any problem. To show the flexibility of the Vlv-ACC, the Gazis-Herman-Rothery (GHR) following model [12] was implemented. This model is a classic car following model for traffic flow modeling.

The Gazis-Herman-Rothery (GHR) car following model is one of the most well-known models and was developed in the early sixties at the General Motor Research labs in Detroit. It is based on the assumption that the driver's acceleration is proportional to the velocity difference and to the following distance. The model is expressed in the simple following formula (3).

$$a\_GHR_n(t) = c v_n^m(t) \frac{\Delta v(t-T)}{\Delta x^l(t-T)} \tag{3}$$

where a_n is the calculated FV acceleration, v_n the driven vehicle velocity, Δv the differential velocity and Δx the distance between vehicles. C, m and l are the driver parameters.

$$a_n(t) = a\_GHR_n(t) + \left(\frac{\Delta x(t-T) - d_{follow} - q}{r}\right) \tanh\left(\left(\frac{\Delta x(t-T) - d_{follow}}{p}\right)^3\right)\left(1 - \left(tanh\left(\frac{\Delta v(t-T)}{s}\right)\right)^4\right) \tag{4}$$

where d_{follow} is the desired average following distance, and p half of the dead-zone size on the relative distance axis, q the offset of the correction curve on the relative distance axis, r the proportional coefficient value on the relative distance, and s represents limiting coefficient based on the relative vehicles velocity.

The first term of the correction formula (4) enables to keep a correct relative distance between the vehicles by modifying the acceleration value. The second term represents the zone of action based on the relative distance, and the third term represents the zone of action based on the relative velocity. The last two terms enable to avoid conflict between the GHR car following model and the relative distance correction during high dynamical phases.

As this model was developed in the sixties, it has been used in numerous studies and several parameter sets have been determined [12] to represent global driver behavior. In order to improve accuracy, we a set of parameters determined by Ozaki in 1993 was used. This set includes different parameters for acceleration and deceleration situations. The additional term in (4) has been added to the equation to force the following model to stay within a certain range of distance behind the leading vehicle. Indeed, the original GHR model cannot correctly handle the leading vehicle relative distance discontinuity. This term is only active during low dynamics phases, if the differential velocity between the vehicles is low. Thus it does not interfere with the GHR model during acceleration and braking phases. The selected distance is 60m, and is a coherent value regarding the driving data collected in section IV.

Variable	c	m	l	d_{follow}	p	q	r	s
Value if $a_n \leq 0$	1.1	0.9	1	60	10	10	50	1
Value if $a_n > 0$	1.1	-0.2	0.2	60	10	10	50	1

Table 1: GHR model values, from Ozaki (1993) model calibration [12]

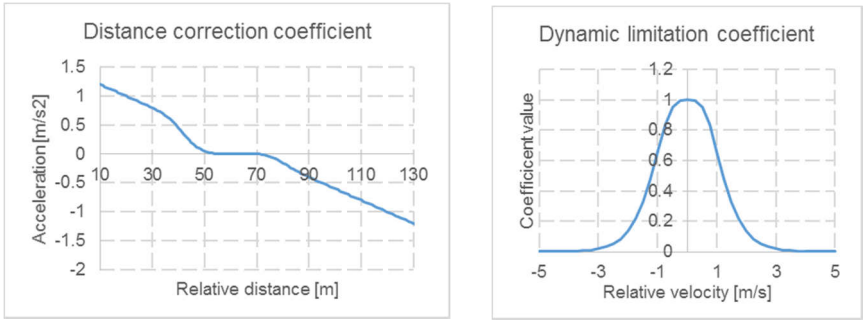


Figure 8: Corrective coefficients to the GHR model.

VI. ACC simulated usage demonstration

This section is dedicated to the usage demonstration of the Vlv-ACC model. Three main driving situations are exposed. The following models are the machine learning PrARX model, explained in section IV, with the previously identified driving parameters, and the modified version of the classic car following model Gazis-Herman-Rothery exposed in section V.
The following driving situations are representing classic use, with “soft” switching cases on figure Figure 9, “hard vehicle in” figure Figure 10, and “hard vehicle out” situation in figure Figure 11.

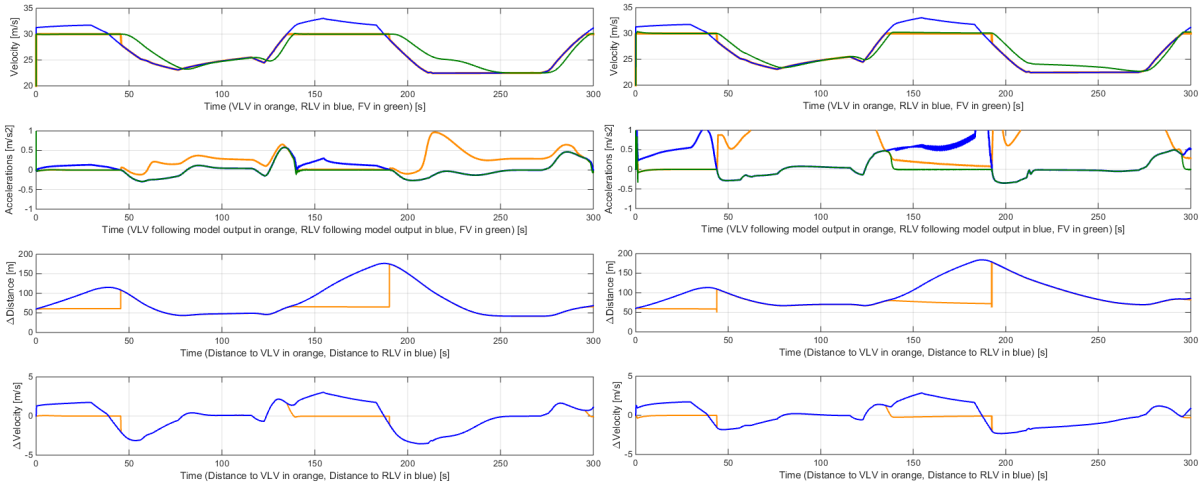


Figure 9: 30 m/s Vlv-ACC example with PrARX car following model on the left, with modified GHR car following model on the right.

We can see on Figure 9 that both following models are able to handle soft switching conditions without any issue. For t=[0, 40] and t=[130, 190] seconds, ACC model is following the virtual vehicle, otherwise,

it is following the real leading vehicle. The lead following vehicle can be understood by looking at the relative distance graph. The model response in green on the acceleration curve differs slightly but the global behavior of the vehicles are similar. There is no major discontinuity due to the lead vehicle switch, the obtained results dynamics can be considered as satisfying.

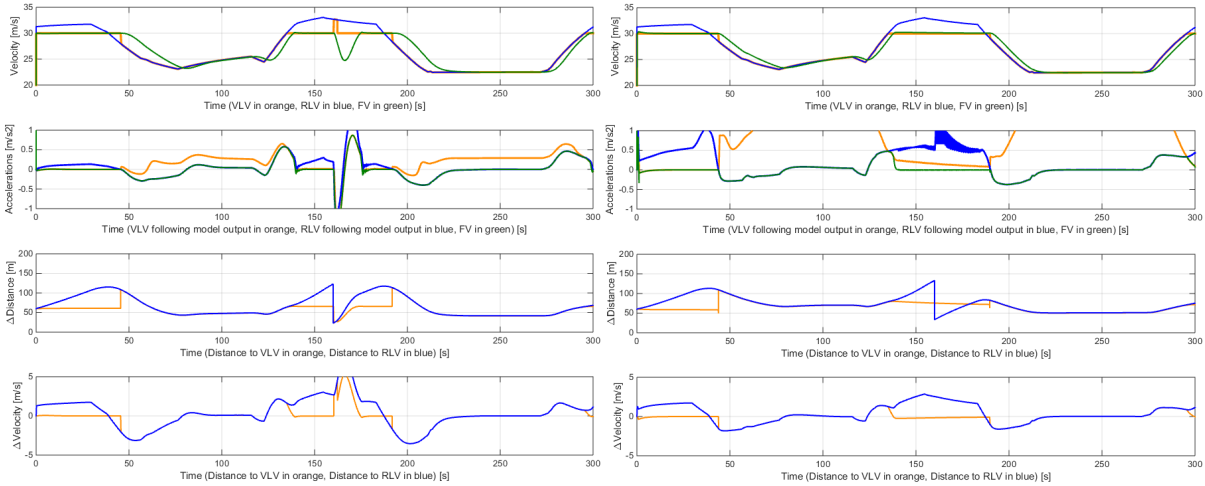


Figure 10: 30m/s Vlv-ACC with PrARX following model on the left, with modified GHR model on the right, 'Hard RLV in' at t=160s.

Figure 10 shows the response of the models to the insertion of a RLV during a VLV following phase (cut-in), at t=160s. We can see that the PrARX based Vlv-ACC responds correctly, by decelerating the vehicle to keep a correct safety distance. In the case of the GHR based Vlv-ACC, we can see that no measure is taken by the model to insure a correct safety distance. It shows that the RLV insertion has not been detected, and the “Hard RLV in” case has not been triggered. This is due to the conception of the GHR model, based on the differential velocity divided by the differential distance. This model cannot handle lead car position discontinuities to adapt the following vehicle behavior. The GHR model is not a good candidate with the selected state switching rules.

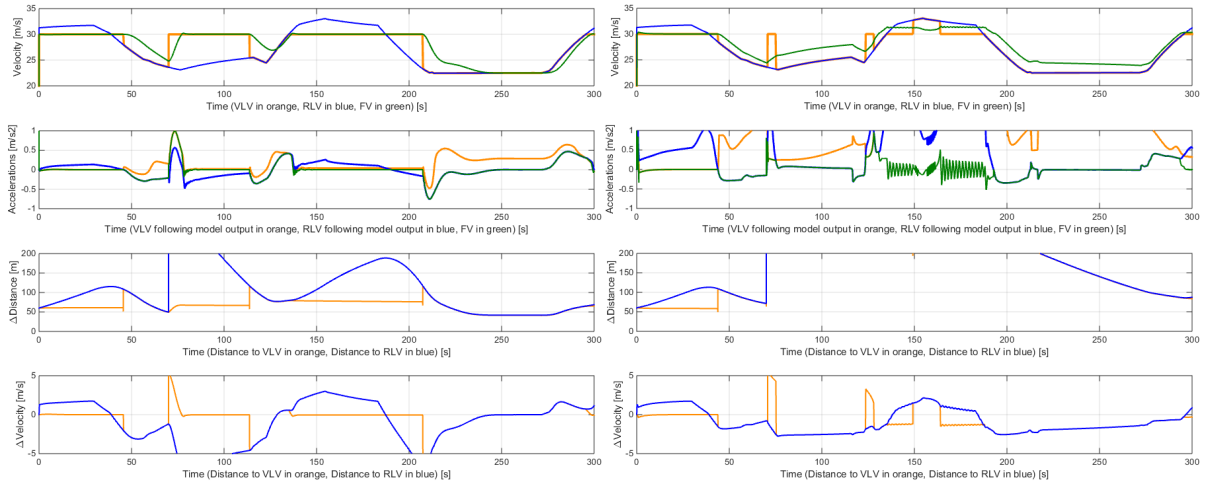


Figure 11: 30 m/s Vlv-ACC with PrARX following model on the left, with modified GHR model on the right, 'Hard RLV out' at t=70s.

Figure 11 shows the exit of a RLV at t=70s. This case can represent the departure of the leading vehicle from the highway. The PrARX based model react in a logical manner, by following the VLV. However the GHM based model does not handle the situation correctly. This is due to the fact than despite the distance separating the following car and the RLV, the model output is lower for the RLV following than the VLV following. Thus the models switches back to RLV following mode and the result becomes meaningless.

We could observe in the previous comparison that the logic switching has to be adapted to the selected following model. The switching logic has been developed based on human reactions. The PrARX car following model can be correctly reproduce human reactions, and therefore lead to a coherent model behavior. Nevertheless, the GHR model is not adapted to this iteration of the Vlv-ACC. Adaptation to the cruise control logic would have to be done to insure correct situation selection.

VII. ACC model comparison

In this section, the comparison between the PrARX based Vlv-ACC and two classic ACC models is done. The goal is to show the main following characteristics of the PrARX model next to industry standard models.

a) Gipps model

The Gipps model is a classic collision avoidance model type [12-13]. It means that this type of model seeks to find a safe following distance to the lead vehicle.

$$\mathrm{v}(\mathrm{t}+\mathrm{t}_{\mathrm{reac}}) = \min \left\{ \begin{array}{c} v(t) + 2.5 a_{max} t_{reac} \left(1 - \frac{v(t)}{v_0}\right) \sqrt{0.025 + \frac{v(t)}{v_0}} \\ b t_{reac} + \sqrt{b^2 t_{reac}^2 + b\left[2(\Delta x(t) + s_0) + v(t) t_{reac} + \frac{v_{lead}(t)^2}{b_{max}}\right]} \end{array} \right\} \quad (5)$$

with $v(t)$ the vehicle velocity, $\Delta x(t)$ the relative distance between vehicles, and the other parameters as follow:

Table 2: Gipps model parameters.

Description	Desired velocity [m/s]	Stop distance to lead vehicle head [m]	Reaction time [s]	Maximal acceleration [m/s2]	Desired deceleration [ms2]	Maximal deceleration [m/s2]
Variable	v_0	s_0	t_{reac}	a_{max}	b	b_{max}
Value	30	5	0.3	1.5	1.5	2.5

The parameters have been attributed so to match the PrARX model behavior.

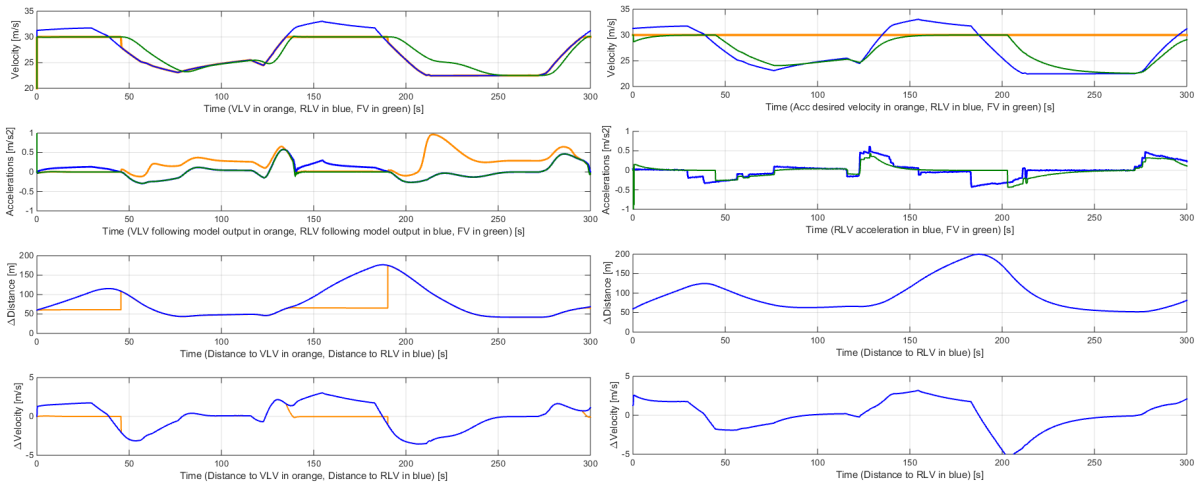


Figure 12: Following situation with the PrARX Vlv-ACC on the left, and with the Gipps model on the right.

We can observe on Fig.17 that the Gipps model represents correctly the driver behavior. The response time, the following distance and the acceleration are close to the PrARX Vlv-ACC. Nevertheless the model output have the disadvantage to be discontinuous. Thus filtering should be applied, implying an undesired increase in the model response time.

b) IDM model

IDM is a time-continuous car-following model, developed to improve older models as Gipps model [14-15]. It has been used as base for the implementation of an ACC in a Volkswagen vehicle in the INVENT project.

$$\dot{v}(s, v, \Delta v) = a\left[1 - \left(\frac{v}{v_0}\right)^4 - \left(\frac{s^*(v, \Delta v)}{s}\right)^2\right]$$
$$s^*(v, \Delta v) = s_0 + vT + \frac{\Delta v}{2\sqrt{ab}} \quad (6)$$

with v the vehicle velocity, Δv the relative velocity with the leading vehicle, s the relative distance with the leading vehicle, and the following parameters, selected to match the PrARX model behavior:

Description	Desired velocity [m/s]	Jam distance [m]	Safety time [s]	Maximal acceleration [m/s2]	Desired deceleration [m/s2]
Variable	v_0	s_0	T	a	b
Value	30	2	2	1.4	2

Table 3: IDM model's selected parameters values

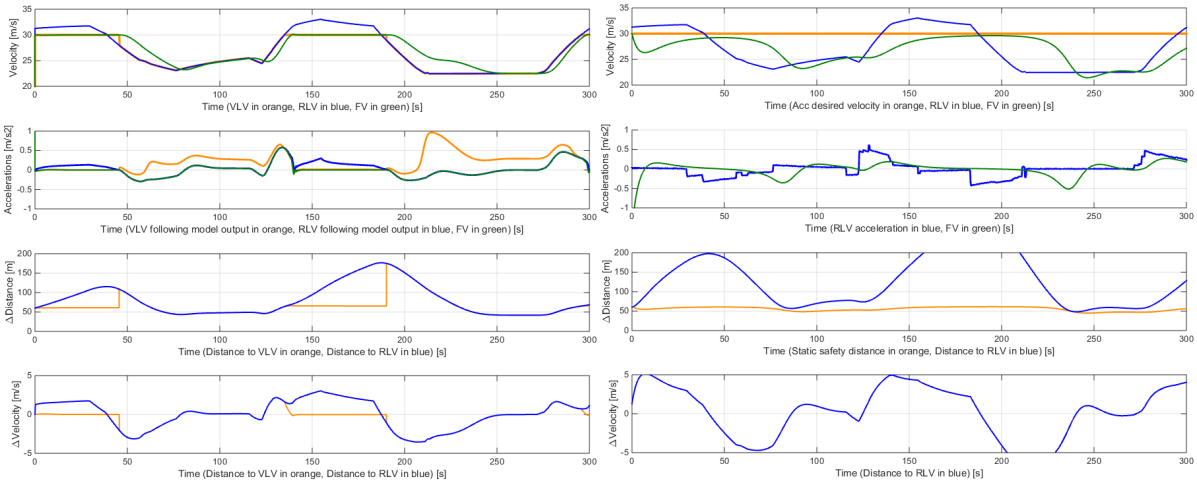


Figure 13 : Following situation with the PrARX Vlv-ACC on the left, with the IDM algorithm on the right. Static safety distance represents the s^* equation without the relative velocity term.

We can observe on Fig. 13 that despite the optimization of the parameters, the IDM model has a very different behavior. The vehicle accelerations are smoother but act later, focusing on the preservation of the correct safety distance. We can clearly see this situation on the [200-250] seconds phase, where the PrARX model slows the vehicle down depending on the distance and relative velocity to the lead vehicle, while the IDM model only acts to reach the ideal safety distance. It is realistic to believe that the early braking behavior of the PrARX Vlv-ACC model enables the driver to understand the behavior of the ACC, and thus to have a good confidence in the system.

VIII. Real world application proposition

In this section, possible implementation in a real vehicle is exposed.

a) Implementation of the PrARX algorithm

To be able to use the PrARX Vlv-ACC, tree main steps have to be followed. At first the driving data has to be measured, then the driver has to be identified, and finally the PrARX model is able to reproduce the driver behavior. Both measurements and reproduction of the driver's behavior are very light processes in terms of computation power and can be executed in a low-power ECU of the car. The driver parameters identification algorithm is heavier, due to a process of optimization using a gradient descent algorithm. Parameter identification can be done on a remote server through network communication. In

case there is a need to adapt parameter's definition online, a low-power version of the algorithm has also been developed [16].
Each iteration of following behavior estimation takes about ten micro-seconds, while the full identification operation can take a few minutes on a modern computer. If the algorithm runs at 10Hz, we see that there should not be any issue with current vehicle computation speed capacity.

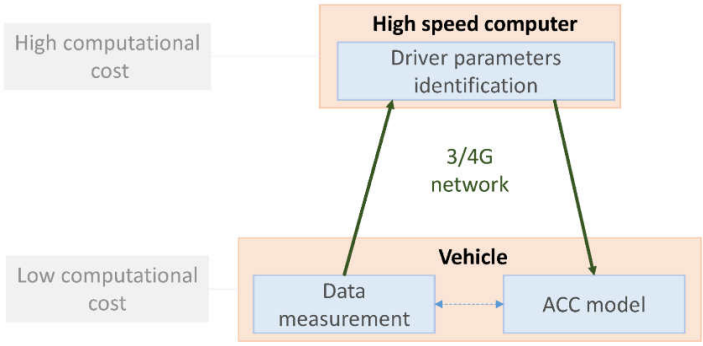


Figure 14: Proposition of vehicle implementation for PrARX model's parameters identification.

b) Implementation of the Vlv-ACC

Various driver models can be used in the Vlv-ACC. ARX models and other autoregressive systems are sensible to information delay. This delay can act as a gain in the closed-loop system or can lead to instability. Moreover, the selected machine learning vehicle following method, PrARX, reproduces the combination of the driver and vehicle behavior. Thus, the driven vehicle should not be in the control loop.
To avoid any of these situations happening, separating the model in two distinct control loops is advised. As shown on Fig. 15, the real vehicle is controlled based on a simulated vehicle. When using the PrARX Vlv-ACC model, the model calculated vehicle is simply the integration for the Vlv-ACC output, with real vehicle dynamical limits (i.e. max accelerations). Thus it can be used to frame the model dynamics, but does not cost any calculation time. The real vehicle is then controlled from the modeled vehicle's position. The required relative distance to the leading vehicle can be corrected to represent the distance between the modeled vehicle and the lead vehicle.

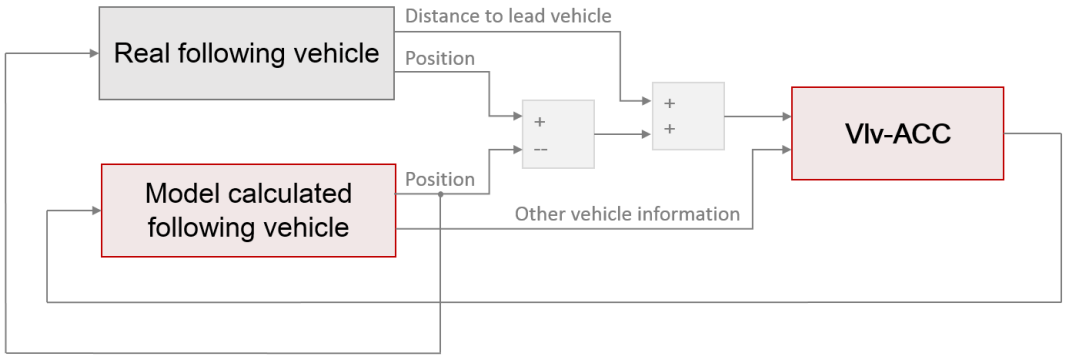


Figure 15: In vehicle Vlv-ACC implementation example.

c) Virtual testing solution using IPG Carmaker/AVL Cruise

In the creation process of the ACC, testing has to be done. This is easily made possible thanks to the use of the two software IPG Carmaker and AVL Cruise. Carmaker in its Matlab Simulink version can recreate the simulation environment, the vehicle sensors, and the vehicle body dynamics. And AVL Cruise can be used to simulate very precisely the vehicle powertrain. Thus, any vehicle including electric, hybrid and conventional combustion engines powertrains can be modeled, with any external road simulation. Moreover using ALV Cruise is a very efficient and precise way to asses and optimize the energy consumption of the ACC system on the implemented vehicle.

IX. Conclusion

In this paper, a new ACC structure able to take advantage of machine learning based vehicle following models was proposed. The concept and the implementation of this structure were exposed. The demonstration of the good functioning of this model was done with the hybrid dynamical PrARX model, a soft switching multi-mode ARX machine learning vehicle following model. The results of the Vlv-ACC model were exposed for different driving situations, including lead vehicle sharp cut-in and lane exit. An important point was underlined: switching case conditions of the system have to be adapted to the selected car-following model. The results were compared to existing vehicle-following and ACC systems, and the PrARX Vlv-ACC showed very competitive behavior. A method to implement the system in a real vehicle, using on-board computation for the ACC program, and cloud-computation for the machine learning driver parameters identification was proposed. Finally, useful tools for model testing were exposed.

X. Bibliography

[1] J. Bengtsson, (2011). *Adaptive Cruise Control and Driver Modeling*, Lund Institute of Technology

[2] A. Rosenfeld, Z. Bereket, C. V. Goldman, S. Kraus, D. J. LeBlanc, O. Tsimori, (2012). *Learning Driver's Behavior to Improve the Acceptance of Adaptive Cruise Control*, IAAI

[3] S. Glaser, B. Vanholme, S. Mammar, D. Gruyer, L. Nouveliere, (2010). *Maneuver-Based Trajectory Planning for Highly Autonomous Vehicles on Real Road With Traffic and Driver Interaction* , IEEE Trans. on Intel. Trans. Systems, Vol. 11, No. 3

[4] L. Zhou, (2015). *The Advance of Intelligent Mobility and Co-Creation of Ecosystems*, Connected Car Japan Keynote session, Automotive World

[5] Charles C.MacADAM, (1981). *Application of an Optimal Preview Control for Simulation of Closed-Loop Automobile Driving*, IEEE Trans. on systems and cybernetics, Vol. SMC-11, No. 6

[6] P. Angkititrakul, C. Miyajima, K. Takeda, (2011). *Modeling and adaptation of stochastic driver-behavior model with application to car following*, IEEE Intelligent Vhl. Symposium (IV), pp. 814-819,

[7] J. Sjöberg, Q. Zhang, L. Ljung, A. Benveniste, B. Delyon, P.Y. Glorennec, H. Hjalmarsson, A. Juditsky, (1995). *Nonlinear black-box modeling in system identification: A unified overview*, Automatica, vol. 31, no. 12, pp. 1691–1724, Dec.

[8] K.S. Narendra and K. Pathasarathy, (1990). *Identification and control of dynamical systems using neural networks*, IEEE Trans. Neural Networks, Vol. 1, No. 1, pp. 4–27

[9] H. Okuda, N. Ikami, T. Suzuki, Y. Tazaki, K. Takeda, (2013). *Modeling and Analysis of Driving Behavior Based on a Probability-Weighted ARX Model*, IEEE Trans. on Intel. Trans. Systems, Vol. 14, No. 1

[10] R. Wade Allen, T. J. Rosenthal, B. L. Aponso, (2005). *Measurement of Behavior and Performance on Driving Simulator*, Driving Sim. Conf, North America

[11] Charles C.MacADAM, (2003). *Understanding and Modeling the Human Driver*, Swets & Zeitlinger, Vehicle System Dyn., Vol. 40, No. 1-3, pp. 101-134

[12] M. Brackstone, M. McDonald, (1990). *Car-following: a historical review*, Transportation research part F, pp.1881-196

[13] J. J. Olstam, A. Tapani, (2004). *Comparison of Car-following models*, VTI meddelande 960A

[14] M. Treiber, A. Hennecke, D. Helbing, (2008). *Congested Traffic States in Empirical Observations and Microscopic Simulations*, University of Stuttgart, Germany

[15] A. Kesting, M. Treiber, M. Schönhof, D. Helbing, (2007). *Extending Adaptive Cruise Control to Adaptive Driving Strategies*, Trans. Research Record: Journal of the Trans. Research Board, No 2000, pp.16-24

[16] N. Ikami, H. Okuda, Y. Tazaki, T. Suzuki, (2011). *Online parameter estimation of driving behavior using probability-weighted ARX models*, IEEE Intel. Trans. Syst., pp. 1874-1879