Final Degree Project

Grau en Enginyeria en Tecnologies Industrials

Simulations of a variable structure-based algorithm for adaptive cruise control

THESIS

Author: Marc Sáez Cobos
Director: Arnau Dòria Cerezo
Date: June 2018

ETSEIB

Escola Tècnica Superior
d’Enginyeria Industrial de Barcelona

UPC

Abstract

The aim of this work is to test in simulations a variable structure algorithm for adaptive cruise control in vehicles to study the influence it may have on traffic flow.

The simulation tests will include two scenarios: first, the platooning problem in vehicles, that consists in to achieve a stable flow for N controlled vehicles and, secondly, the traffic flow behaviour when this ACC algorithm is implemented in an heterogeneous fleet mixing human drivers and autonomous vehicles.

To accomplish this purpose, tools as *Simulink* provided by *Matlab software* have been used as well as their own programming language.

A traffic model will be designed and the variation in platooning and string stability will be studied in the cases mentioned above, making variations in the parameters of the model.

Resum

L'objectiu d'aquest treball és provar en simulacions un algoritme d'estructura variable per a control adaptatiu de creuer en vehicles per veure la influència que pot tenir sobre el trànsit.

Les proves de simulació inclouran dos escenaris: en primer lloc, el problema de “platooning” en vehicles, que consisteix a aconseguir un flux estable per a N vehicles controlats i, en segon lloc, el comportament del flux de trànsit quan aquest algoritme ACC s'implementa en una flota heterogènia que barreja conductors humans i vehicles autònoms.

Per dur a terme aquest propòsit s’empren eines que proporciona el software *Matlab*, com *Simulink* així com el seu propi llenguatge de programació.

Es crearà així un model de trànsit i s’estudiarà la variació en el platooning i string stability en els casos abans mencionats fent variacions en els paràmetres del model.

Summary

List of figures

List of tables

1. Glossary

ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance Systems
CACC	Cooperative adaptive cruise control
IDM	Intelligent Driver Model
SAE	Society of Automotive Engineers
V2I	Vehicle capable to communicate with infrastructures
V2V	Vehicle capable to communicate with other vehicles

2. Preface

2.1. Project Context

Traffic congestion is a major problem which bothers our urban traffic sustainable development at present. These phenomena are multifaceted, and it depends on the interpretations of many vehicles. Vehicles also do not react simply following the laws of mechanics, they vary depending on individual driver's behavior.

Congestion phenomenon cannot be terminated only by applying physical constructions to increase road capacity. Although the large quantity of drivers is certainly the most important factor, another relevant one is the lack of coordination between them. It is necessary to develop different technologies in order to control the traffic phenomenon and reproducing real traffic with simulations is one way to deal with the problem.

Traffic flow models are developed to reproduce the characteristics of traffic flow by various modeling methods. It is normally constrained along a one-dimensional pathway, and could be classified into macroscopic, mesoscopic, and microscopic models. All the models have their own application areas but all of them will help to analyze which factors are determining in real traffic. Once the most important factors are determined, it is possible to build algorisms to improve traffic congestion. Moreover, as it seems impossible to change the population driving habits, implementing these algorisms in autonomous driving is an effective measure to improve traffic flow.

Thus, autonomous driving consists in incorporate motion control algorithms in vehicles giving them the ability to drive themselves without human assistance.

2.2. Project origin

Currently, the *Institut d'Organització i Control de Sistemes Industrials (IOC)* of the *Universitat Politècnica de Catalunya (UPC)*, is working in an important project of developing line follower robots. They are working on different ways to upgrade them and several have already been tested.

This project is oriented to the implementation of a vehicle tracking controller, given by the IOC, that is able to adapt the speed of the vehicle to run at a desired speed and to avoid collisions. Furthermore, there is the intention of improving traffic flow. The implementation of the controller in the robots is not considered in the project, but it is a possible goal for future projects.

3. Introduction

3.1. Objectives

This project aims to provide evidence of how autonomous driving (and in particular the Adaptive Cruise Control) could be an effective measure to alleviate urban traffic congestion. For this purpose, software MATLAB and SIMULINK are used. The following objectives are defined for the project:

- Study and test by means of simulations of the ACC controller given [1] .
- Simulate human driving behaviour with the IDM model.
- Build a simulation platform to test the influence of autonomous driving in real traffic with the capability of easy changing different model parameters. And, not very thoroughly, detect the phenomena of platooning, string stability and stop and go.

3.2. Scope of the project

The first part of the project is related to the ACC for autonomous vehicles given [1]. It will be studied in detail and tested in simulations to test its behaviour.

Secondly, it will be necessary to model the human driving behaviour and, to compare results with the previous model, the same simulations will be set.

Finally, to test the interaction between them, a simulation platform will be drawn, with the objective of having it with the facility of easily changing the simulation parameters involved.

It should be said that all simulations, for example, do not allow lane changing or negative velocities.

4. The ACC controller

Advanced Driver Assistance Systems (ADAS) have lately drawn a lot of research interest in order to improve safety, traffic flow and energy efficiency. The use of on-board sensors allows the vehicle to communicate with other vehicles (V2V) or with infrastructures (V2I). One of the most representative ADAS is the Adaptive Cruise Control (ACC) which corresponds to Level 2 of SAE (Society of Automotive Engineers) levels of driving automation. This graduation goes from no-automation (Level 0) to full automation (Level 5).

The ACC is usually simplified into a platooning problem, where the objective is to regulate the distance with the precedent vehicle and by cooperation among autonomous vehicles achieve a stable flow, including string-stability behaviour [2]- [3]. Thus, to reach that goal, the speed of the autonomous car is modified according to a field variable.

In this case, the ACC given is described and tested in [1]. It is based on a variable structure-based algorithm which main feature is that provides two different driving modes, named speed and distance modes, and achieved using a switching surface ruled by a distance. In one hand, the speed mode or cruise control regulates velocity to induce sliding motion when the aimed cruise velocity of the target vehicle is higher than the velocity of the preceding vehicle. In the other hand, when it is lower, the distance mode is in charge of keeping a safe following distance among cars by asymptotic tracking of the cruise velocity.

4.1. Model

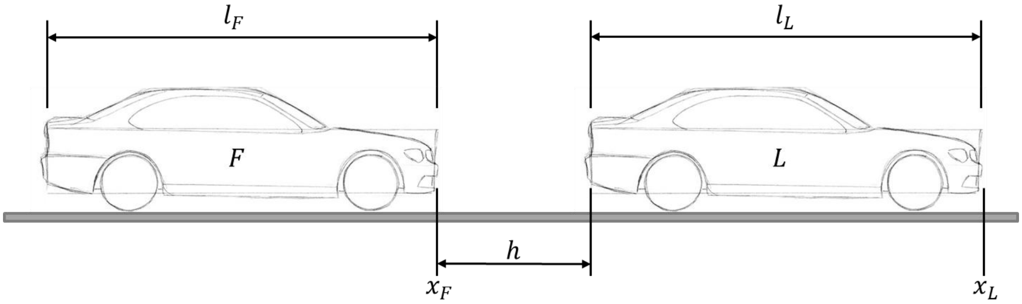


Figure 1: Two vehicles (leader (L) and follower (F)) along the same path.

Consider two vehicles (a leader, L, and a follower, F) moving along the same path, see Figure. 1. The distance between them is given by:

$$h(t) = x_L(t) - x_F(t) - l_L \qquad \text{(eq.1)}$$

where x_L , x_F are the absolute positions of the leader and follower vehicles, respectively, and l_L is the length of the leader vehicle. The follower vehicle is modelled as a mass m moving along a straight line at velocity v,

$$\dot{x}_F = v, \qquad \textit{(eq.2)}$$

$$m\dot{v} = -F_c - F_r - F_d(v) + u \qquad \textit{(eq.3)}$$

where u is the acceleration/braking force applied by the engine, and $F_c,\ F_r,\ F_d(v)$ represent the climbing, rolling and aerodynamic forces, respectively.

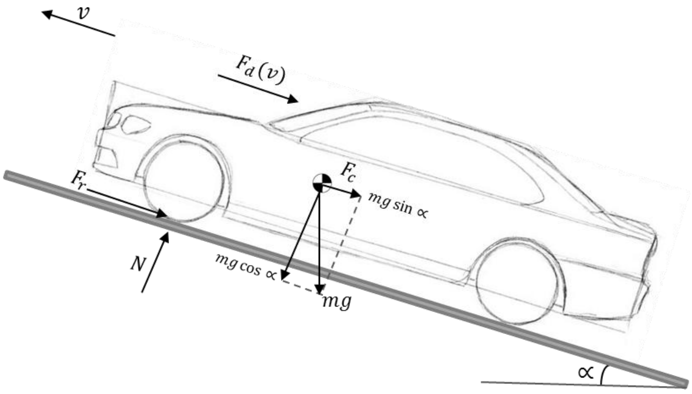


Figure 2: Existing forces on the vehicle

The forces $F_c,\ F_r,\ F_d(v)$ are modelled, according to Figure 2, as:

$$F_c = mg \sin \propto \qquad \textit{(eq.4)}$$

$$F_r = f_r mg \cos \propto \qquad \textit{(eq.5)}$$

$$F_d(v) = \frac{1}{2}\rho C_d A v^2 \qquad \textit{(eq.6)}$$

where g is the gravity, $\propto$ is the road slope angle, f_r is the rolling coefficient, ρ is the air density, C_d is the drag coefficient and A is the vehicle frontal area.

Combining (1) with (2)-(3) we get

$$\dot{h} = -v + v_L \qquad \textit{(eq.7)}$$

$$m\dot{v} = -F(v) + u \qquad \textit{(eq.8)}$$

where $v_L = \dot{x}_L$ is the leader velocity and $F(v)$ is the sum of forces against movement

$$F(v) = F_c + F_r + F_d(v) \qquad \textit{(eq.9)}$$

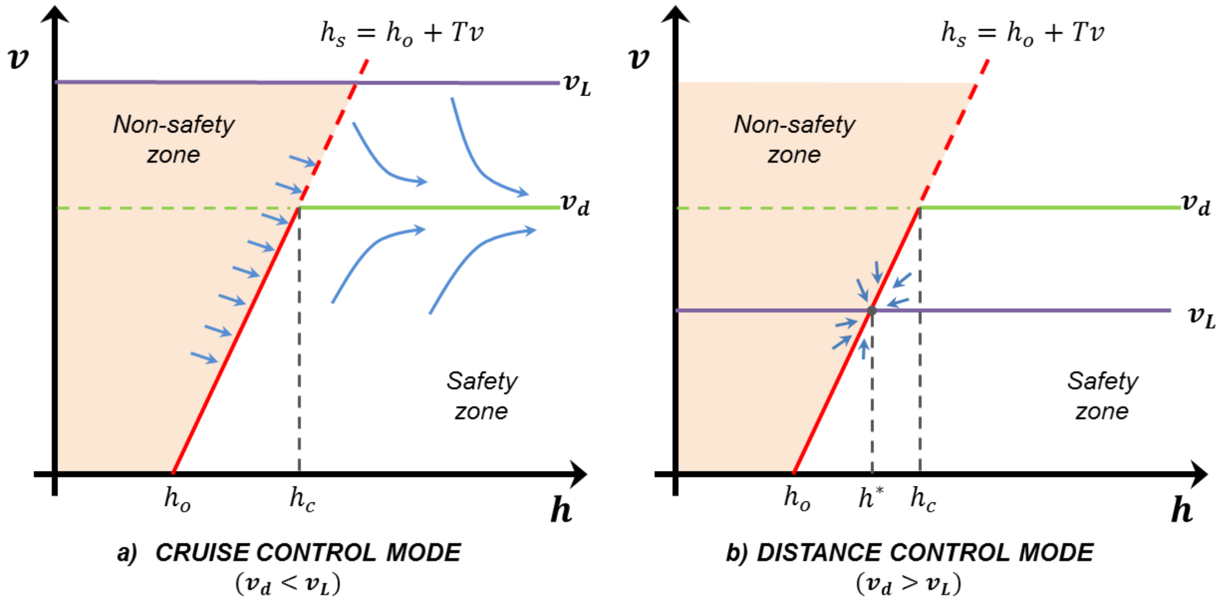
4.2. Problem formulation

Roughly speaking, the aim of the ACC algorithm is twofold: to regulate the velocity to a desired value, v_d, but at the same time, guaranteeing a safety distance. In this case, it depends on the speed of the vehicle. Accordingly, the following distance policy is adopted:

$$h_s = h_o + Tv \qquad \textit{(eq.10)}$$

where h_o is the minimum desired distance to the precedent vehicle, and T is the desired safety time headway when following other vehicles (the time it takes the vehicle to arrive to the position of its predecessor). Thus, to guarantee the desired safety the relation $h < h_s$ must be fulfilled.

In order to evince the behaviour of the two different modes switching control, the study of the plane (h, v) is shown in Figures 3a and 3b.



a) CRUISE CONTROL MODE $(v_d < v_L)$ **b) DISTANCE CONTROL MODE** $(v_d > v_L)$

Figure 3: Control objectives for each ACC control mode

Assuming v_L constant, the two control modes are represented in Figures 3a and 3b, with the desired speed v_d (in green), the minimum distance h_s (in red) and the non-safety zone (in orange). The problem can be initially stated by defining two behaviours:

a) Cruise control: regulates the speed to a desired value, v_d, when $v_d < v_L$. Consequently, the control makes the speed reach the desired value $(v \to v_d)$ and using the equation (7) one gets that the distance h increase with time (see the blue trajectories in the safety zone in Figure 3a). The achievement of a safety distance is by far guaranteed.

b) Distance control: when the leader vehicle runs slower than the desired cruise control speed, $v_d > v_L$, the objective is to regulate the distance with the precedent vehicle to the safety distance given in (10). Then, as v_L is constant, the equilibrium of (7) implies that the vehicle will tend to run at the same speed as the leader $(v \rightarrow v_L)$. Consequently, the target distance is:

$$h^* = h_o + Tv_L \qquad \textit{(eq.11)}$$

that corresponds to the intersection of (10) with v_L (it can be noticed in Figure 3b). Also in Figure 3b the blue trajectories represents the desired behaviour, stabilizing at (h^*, v_L) moving out from the non-safety zone.

4.3. Variable structure controller design

To simplify the model, it can be assumed that the sum of forces against movement term, $F(v)$, is known in (8). This simplification allows modelling the control action as:

$$u = u_C + u_V \qquad \textit{(eq.12)}$$

with $u_C = F(v)$.

In addition, from the discussion in 4.2, it seems reasonable to propose the following switching action for u_V:

$$u_V = \begin{cases} f_1(h,v) & s > 0, \\ f_2(h,v) & s < 0. \end{cases} \qquad \textit{(eq.13)}$$

where f_1 and f_2 are designed as:

$$f_1(h,v) = -k_0^v(v - v_d) \qquad \textit{(eq.14)}$$

$$f_2(h,v) = \begin{cases} k_1^h s - k_1^v v & v > 0, \\ 0 & v = 0. \end{cases} \qquad \textit{(eq.15)}$$

and the switching function being:

$$s(h,v) = h - h_0 - Tv \qquad \textit{(eq.16)}$$

See [1] for a more detailed stability discussion.

5. Simulations of the ACC controller

To prove the ACC controller works properly it is necessary to test by numerical simulations. To be able to do it, an appropriate software must be chosen.

To implement the equations of the model and to represent the simulation graphically, it is necessary to find a software that satisfactorily met the requirements. Given that in the model there are differential and integral equations, which some of them are non-linear, and due to the difficulty to analytically deal with them, the software must have tools capable of provide numerical integration. Of all the available softwares *Matlab software* has been chosen.

Matlab is a software of great capacity for calculation and is widely used throughout the world, whether in research at universities or developing products in companies. The great documentation and online community are really helpful to carry out the project and find solutions to the problems that have arisen throughout this one.

In addition, the software incorporates a simulation platform called *Simulink*, ideal for our purpose. It is a very interesting tool for simulate dynamic systems and its visual programming environment based on blocks allows to implement the equations of any system in a simple and intuitive way.

5.1. Leader and ACC follower

The first simulation test consists in both vehicles starting at zero speed ($v(0) = v_L(0) = 0km/h$) and with an initial distance of $70m$ between them. Then, the leader vehicle, starting at $0km/s$, accelerates to $80km/h$ and brakes to $25km/h$ every $20s$ (with a first order behaviour where the time constant is $\tau = 1{,}2s$). On the contrary, the ACC vehicle will tend to achieve the desired speed of $v_d = 70km/h$.

As a result, the follower vehicle is forced to adapt its speed to the leader vehicle, when the leader vehicle breaks, and to achieve the desired speed when the leader accelerates. This causes the controller to work through the two different operating modes. That will allow checking if the two driving modes work correctly.

5.1.1. Parameters used

In this case, it has been considered that the two vehicles have the same parameters (see Table 1).

Vehicle parameter	***Value***	***Vehicle parameter***	***Value***
Vehicle mass, m	$1000\ kg$	Minimum desired distance, h_0	$2\ m$
Gravity, g	$9{,}8\ m/s$	Time headway, T	$1{,}7\ s$
Rolling coefficient, f_r	$0{,}0017$	k_0^v	588
Air density, ρ	$1{,}225\ kg/m^3$	k_1^h	600
Drag coefficient, C_d	$0{,}3$	k_1^v	100
Vehicle frontal area, A	$2{,}8\ m^2$	Initial distance, h_{in}	$70\ m$
Road slope, $\propto$	0°	Initial speeds, $v = v_L$	$0\ km/h$

Table 1: Parameters used during the leader ACC follower simulation (extracted from [1])

5.1.2. Simulation structure

Some simulations are carried out to test that the ACC controller works as expected. As it was mentioned before, with Simulink tool provided by *Matlab software.*

All the parameters will be introduced in a *Matlab (.m)* file, where, through the results obtained with the Simulink simulation, the necessary plots will be drawn to check the ACC control behaviour. See *ANNEX B.1*.

Using Simulink the following structure will be set:

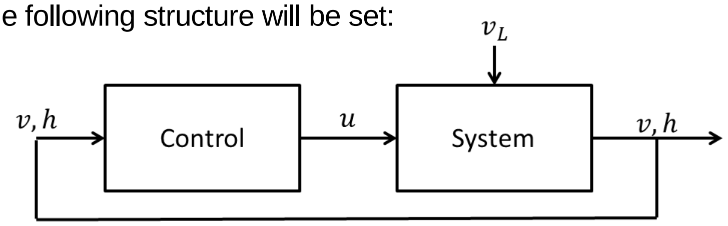


Figure 4: Scheme of the Simulink blocks in the leader and follower

a) System block: is in charge of enforcing the equations that characterize the dynamic model of study (Chapter 4.1 equations). As it can be noticed in Figure 4, this block uses the acceleration/breaking force outgoing from the ACC control and the incoming leader speed to calculate, through the mentioned equations, the speed of the autonomous vehicle and its distance to the leader vehicle.
b) Control block: the control block characterizes the ACC control using Chapter 4.3 equations. With the speed and distance provided by the system block, recalculates the acceleration/breaking force needed and it sends it back to the system block.

For more detailed information about the Simulink blocks check *ANNEX A.1.*

5.1.3. Simulation results

During this chapter, a plot of the simulation (see Figure 5), using the previous structure and parameters, is commented.

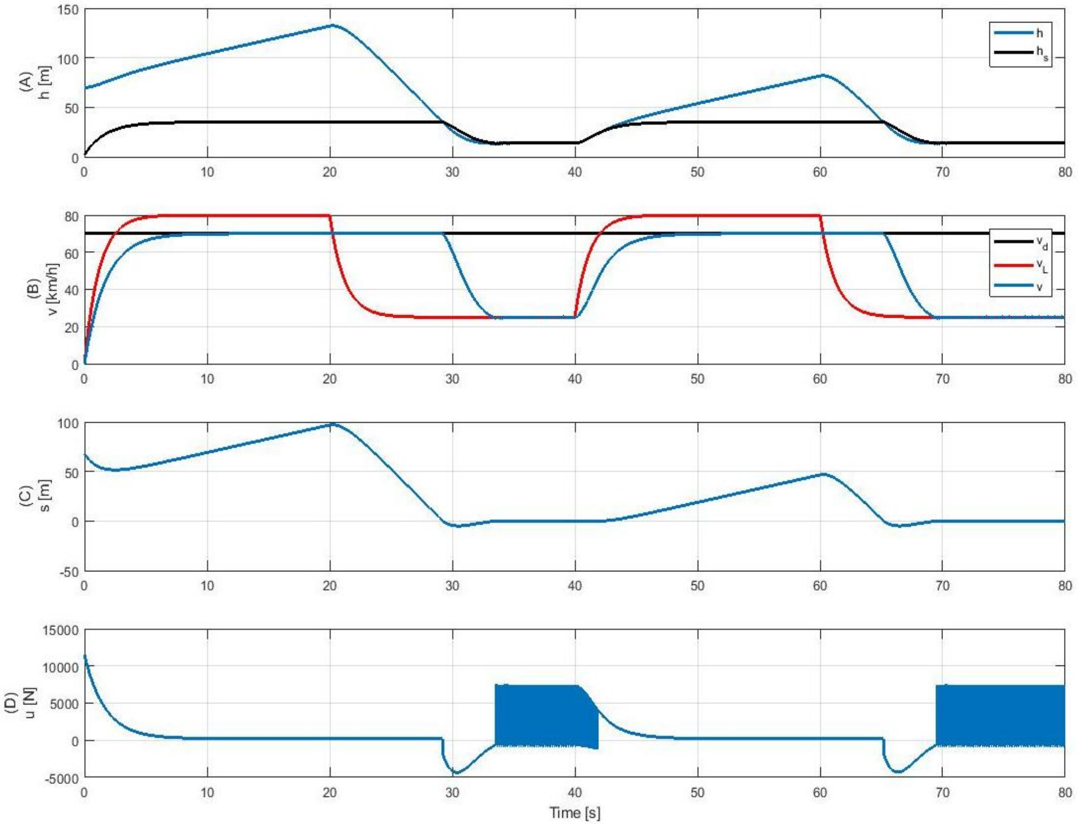


Figure 5: Plots for leader and follower simulation

In the first $20s$, according to Figure 5, both vehicles (the leader and the autonomous) begin to accelerate to acquire their desired speeds of $80km/h$ and $70km/h$ respectively. During these instants $\boldsymbol{v_d} < \boldsymbol{v_L}$ so the ACC control is set to cruise control mode. As we had said before, in this mode the goal is to achieve the desired speed and this is accomplished. As the speed of the autonomous vehicle is slower as the leader one, the distance between them increase as it was expected.

Between $20s$ and $40s$, the leader vehicle brakes till $25km/h$ and remains at this speed for a while. The autonomous vehicle, firstly, keeps running at the desired speed of $70km/h$, and later on breaks to $25km/h$, adapting its speed to the leader one. During this time $\boldsymbol{v_d} < \boldsymbol{v_L}$ so it is important to note that the distance between vehicles is decreasing during this period of time till the switching variable s sticks to zero (here it is noticeable the sliding motion behaviour). At this point the ACC control switches to distance mode. The goal in this control mode is to ensure a minimum safety distance (h^*) so the autonomous vehicle tends to run at the same speed of the leader to stay at this minimum secure distance.

From this moment on, the behaviour during the rest of the simulation repeats again. From $40s$ to $60s$, both vehicles accelerate till achieve their desired speeds and from $60s$ to $80s$ the leader brakes again. The follower adapts its velocity to the leader one to guarantee the minimum safety distance.

From this simulation it can be extracted that the ACC control of the autonomous vehicle behaves as we expected. Although, just in case and to study the phenomenon of string stability, another simulation it is set, this time with N autonomous vehicles running after the leader vehicle.

5.2. Leader and N ACC followers

This second simulation test consists in the same study that in the first one but adding N vehicles after the leader vehicle.

All vehicles again start at zero speed ($\sum_{i=1}^{N} v_i = v_L = 0km/h$) and the distance between them it is initially of $70m$.

Again the leader vehicle starting at $0km/s$, accelerates to $80km/h$ and brakes to $25km/h$ every $20s$, with a first order behaviour where the time constant is $\tau = 1{,}2s$. On the contrary, the ACC vehicles, will tend to achieve the desired speeds of $v_d = 70km/h$.

As a result, the follower vehicles are forced to adapt their speeds to the previous vehicle, when the leader vehicle breaks, and to achieve the desired speeds when the leader accelerates. This causes the controllers to work through the two different operating modes and will allow to study if the string stability is maintained or, on the contrary, if the error is transmitted and in the worst case amplified.

5.2.1. Simulation structure

As the previous simulation, all the parameters will be introduced in a *Matlab (.m)* file, where, through the results obtained with the Simulink simulation, the necessary plots will be drawn to check the ACC control behaviour. See *ANNEX B.2*.

Using Simulink the following structure will be set:

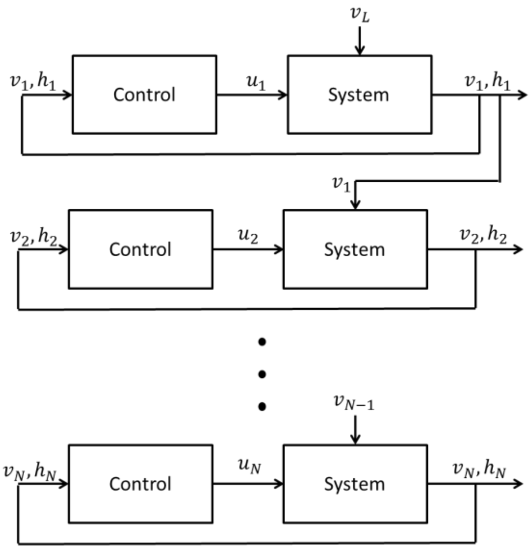


Figure 6: Scheme of the Simulink blocks in the leader and N followers simulation

It is built with the same idea of the previous simulation, this time taking the output velocity of the previous vehicle as the leader velocity.

Notice that, from Figure 6, only the first autonomous vehicle takes the velocity of the leading one (v_L) as a reference. The following autonomous vehicles take the reference of the speed of the previous vehicle.

In this case it will be studied a case where $N = 3$. This means a total of four vehicles, the leader and three autonomous followers, named from now on ACC_1, ACC_2 and ACC_3.

For more detailed information about the Simulink blocks check *ANNEX A.2.*

5.2.2. Parameters used

The following parameters are used:

Vehicle parameter	***Value***	***Vehicle parameter***	***Value***
Vehicle mass, m	$1000\ kg$	Minimum desired distance, h_0	$2\ m$
Gravity, g	$9{,}8\ m/s$	Time headway, T	$1{,}7\ s$
Rolling coefficient, f_r	$0{,}0017$	k_0^v	588
Air density, ρ	$1{,}225\ kg/m^3$	k_1^h	600
Drag coefficient, C_d	$0{,}3$	k_1^v	100
Vehicle frontal area, A	$2{,}8\ m^2$	Initial distances, $h_{in}^{L-1}, h_{in}^{1-2-3}$	$70\ m$ / $80\ m$
Road slope, $\propto$	0°	Initial speeds, $v_1 = v_2 = v_3 = v_L$	$0\ km/h$
Vehicle length, l	$4\ m$		

Table 2: Parameters used during the leader and N ACC followers simulation (extracted from [1]).

The exact same parameters have been used as in the first simulation. However, in future simulations a function will be created that will provide us different parameters according to the average and standard deviation entered. In this way it is possible to get results more closely to real ones.

5.2.3. Simulation results

Using the previous structure and parameters see Figure 7 (plot of the results).

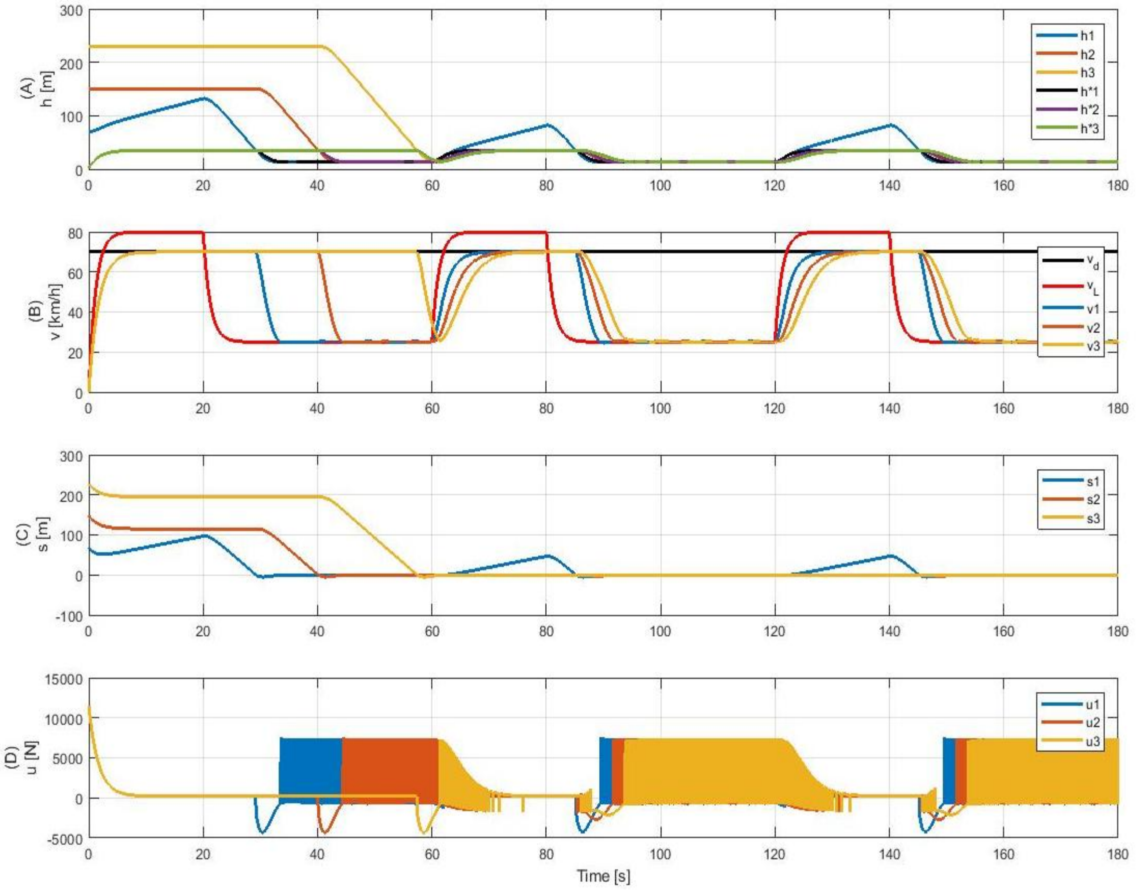


Figure 7: Plots for leader and N autonomous followers simulation $(N = 3)$

In the first $20s$, all vehicles (the leader and the autonomous) begin to accelerate to acquire their desired speeds of $80km/h$ and $70km/h$ respectively. During these instants $\boldsymbol{v_d} < \boldsymbol{v_L}$ so the ACC controls are set to cruise control mode. Notice that the behaviour of the leader and ACC_1 is exactly the same as the first simulation (as the speed of ACC_1 is slower as the leader one, the distance between them increase), unlike ACC_2 and ACC_3 that seem to behave otherwise than the previous ones.

Since the autonomous vehicles have the same properties, the three vehicles initially accelerate in exactly the same way (see Figure 7b), and maintaining then the distance with the previous vehicle as the initial values of $150m$ and $230m$.

Between the $20s$ and $60s$, the leader brakes till $25km/h$ and remains at this speed for a while. ACC_1 , firstly, keeps running at the desired speed of $70km/h$, and later on breaks to $25km/h$, adapting its speed to the leader one. During this time $v_1 < v_L$ so it is important to note that the distance between the leader and ACC_1 is decreasing till the switching variable (s) of ACC_1 sticks to zero, where the ACC_1 control switches to position mode and the sliding motion behaviour occurs $(s_1 = 0)$. As this happens, ACC_2 is approaching ACC_1 , $v_1 < v_2$, until the switching variable s_2 also sticks to zero. At this point ACC_2 drops its speed from the desired to $25km/h$ adapting to ACC_1. Exactly in the same way, the distance between ACC_2 and ACC_3 decreases because $v_2 < v_3$ until $s_3 = 0$ and ACC_3 also brakes to $25km/h$. Therefore, during this time, all ACC vehicles adapt their own speed to the precedent vehicle and, as Figure 7a shows, the minimum safety distances (h_1^*, h_2^*, h_3^*) are fulfilled.

From this moment on, the behaviour during the rest of the simulation repeats again. From $60s$ to $80s$ and from $120s$ to $140s$, all vehicles accelerate till achieve their desired speeds and from $80s$ to $120s$ and from $140s$ to $180s$ the leader brakes again. The followers adapt their velocities to the leader one to guarantee the minimum safety distances.

The objective of this simulation lies in checking if there is string stability or there is an amplification of the error made by the controller. Taking a look at the (s, t) plot, Figure 7c, and zooming around the $30s$, it is possible to notice that when s_1 firstly cross zero, it doesn't stick to zero immediately, but for an instants ACC_1 has a small overshoot where a minimum error is made.

Seconds later ACC_2 and ACC_3 commit also this overshoot. Comparing them, it can be said that those overshoots in ACC_2 and ACC_3 vehicles are of the same magnitude or even smaller as the overshoot of ACC_1.

Taking this in consideration, from this simulation, it can be extracted that the ACC control of the autonomous vehicles has a good behaviour in terms of string stability.

It is important to notice that all the results given by the ACC controller are quite close to an ideal situation where all the vehicles in the road are autonomous-driven. But this is currently away from reality. The autonomous vehicles certainly would have to interact with vehicles driven by humans, so it is necessary to recreate human driving behaviour first to check the real impact of autonomous cars introduced in real traffic flow.

6. Simulation of the IDM model

One of the main objectives of the project is to test out if the incorporation of autonomous vehicles in the road makes possible to improve the traffic flow. To test that, it is necessary to simulate human driving behaviour.

Human driver behaviour has been largely studied. There are several algorithms that provide us an approximate response to how humans control the longitudinal speed of a vehicle depending on the distance to its precedent vehicle. These are known as car following models. The IDM model is one of the most commonly used.

6.1. IDM model

The *intelligent driver model* (IDM) is a time-continuous car-following model for the simulation of freeway and urban traffic. It was developed in [4] to improve upon results provided with other "intelligent" driver models. It describes the dynamics of a single human-drive vehicle.

Based on Figure 1, the dynamics of the ith vehicle including the time derivative of the distance, h_i, is governed by:

$$\dot{h}_i = v_{i-1} - v_i, \tag*{(eq.17)}$$

$$\dot{v}_i = a_i, \tag*{(eq.18)}$$

$$\tau_i \dot{a}_i = -a_i + u_i + d_i, \tag*{(eq.19)}$$

where $v_i = \dot{x}_i$ and a_i are the velocity and acceleration, respectively, τ_i is a time constant, and d_i denotes an additive disturbance. Finally, u_i is the acceleration demand, which depends on the type of the driving that the car is undergoing, namely human controlled. For the sake of simplicity, d_i is assumed to be zero.

Assuming only the longitudinal variables of traffic, the IDM for the ith vehicle is described by (17)-(19), with

$$u_i = a_i^{max}\left(1 - \left(\frac{v_i}{v_i^o}\right)^{\delta_i} - \left(\frac{h_i^*}{h_i}\right)^2\right) \tag*{(eq.20)}$$

where

$$h_i^* = h_i^o + \max\left(0, v_i T_i + \frac{v_i \Delta_{vi}}{2\sqrt{a_i^{max} b_i^{max}}}\right) \qquad \text{(eq.21)}$$

and the variables are:

- v_i is the speed of the ith vehicle.
- a_i^{max} is the maximum acceleration in everyday traffic.
- b_i^{max} is the maximum comfortable deceleration (braking).
- T_i is the desired safety time headway when following other vehicles (the time it takes the ith vehicle to arrive to the position of its predecessor).
- v_i^o is the desired speed when driving in a free road.
- h_i^o is the minimum desired net bumper-to-bumper distance to the precedent vehicle.
- Δ_{vi} is the human perceived relative speed of the ith vehicle with its precedent one.
- δ_i is the acceleration exponent that adds information about how fast the driver accelerates towards its desired speed.

In this case it is also useful to define the error made by the driver as:

$$e_i = h_i - h_i^* \qquad \text{(eq.22)}$$

6.2. Simulation structure

All the parameters will be introduced in a *Matlab (.m)* file, where, through the results obtained with the Simulink simulation, the necessary plots will be drawn to check the IDM control behaviour. See *ANNEX B.3*.

To test this control mode the same structure as figure 6 is done. In this case it will be studied a case, as before, where $N = 3$. This means a total of four vehicles, the leader and three followers, named from now on IDM_1, IDM_2 and IDM_3.

For more detailed information about the Simulink blocks check *ANNEX A.3.*

6.3. Parameters used

The following parameters are used in this simulation.

Vehicle parameter	***Value***	***Vehicle parameter***	***Value***
Vehicle mass, m	$1000\ kg$	Minimum net distance, h_i^o	$2\ m$
Gravity, g	$9{,}8\ m/s$	Safety time headway, T_o	$0{,}7\ s$
Rolling coefficient, f_r	$0{,}0017$	Desired (free road) speed,v_i^o	$60\ km/h$
Air density, ρ	$1{,}225\ kg/m^3$	Maximum acceleration, a_i	$1\ m/s^2$
Drag coefficient, C_d	$0{,}3$	Maxim deceleration, b_i	$3{,}5\ m/s^2$
Vehicle frontal area, A	$2{,}8\ m^2$	Acceleration exponent, δ_i	4
Road slope, $\propto$	0°	Initial speeds, $v_1 = v_2 = v_3 = v_L$	$0\ km/h$
Vehicle length, l_i	$4\ m$	Initial distances, $h_{in}^{L-1}, h_{in}^{1-2-3}$	$70\ m\ /\ 80\ m$

Table 3: Parameters used during the leader and IDM followers simulation (extracted from [1] and [5]).

6.4. Simulation results

From the results of this simulation, the following plot (Figure 8) is presented.

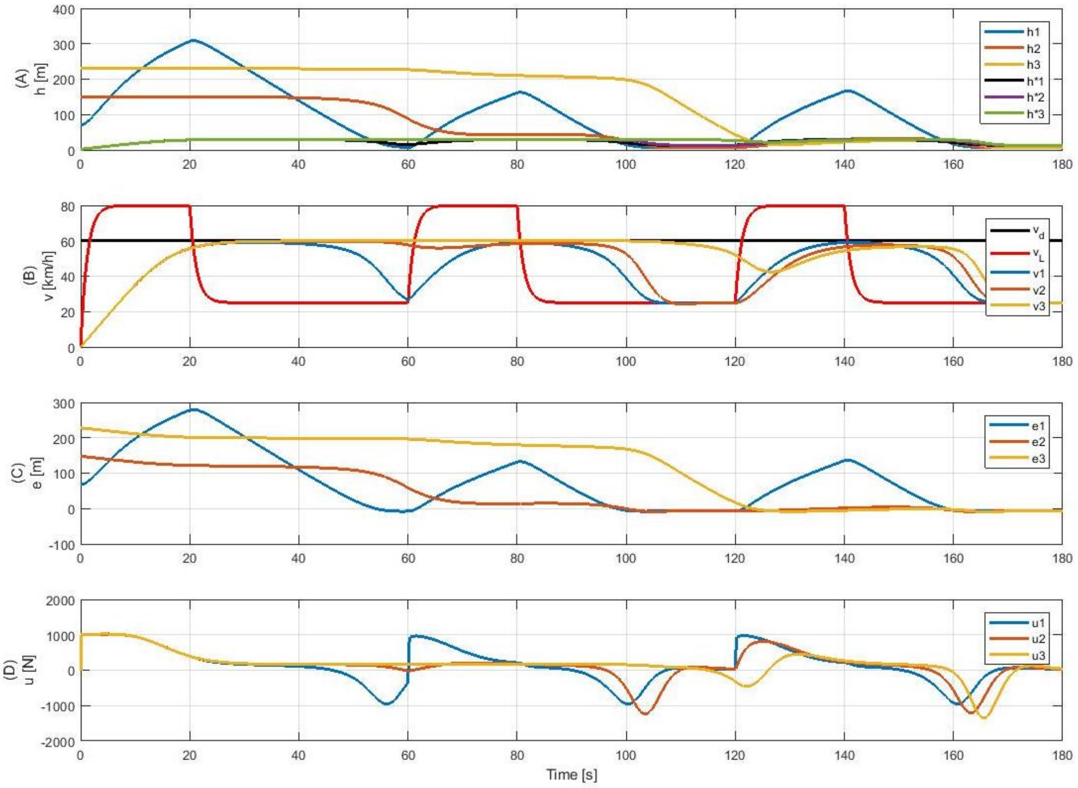


Figure 8: Plots for leader and N IDM followers simulation $(N = 3)$

In this case, as the 5.2 chapter results, the leader accelerate $20s$ till $80km/h$ (the first time from zero, and the following from $25km/h$), and then brakes during $40s$ till $25km/h$.

During the two first sequences (from $0s$ to $60s$ and from $60s$ to $120s$, respectively) the ACC vehicles are catching the leader one so there are not important results.

At the first sequence only IDM_1 approaches the leader and brakes almost to the leader vehicle velocity (the leader accelerates again before IDM_1achieves the $25km/h$). IDM_2 and IDM_3 keep running with their desired speeds while approaching the other vehicles.

At the second sequence IDM_1 and IDM_2, as the ACC vehicles from the previous simulation mentioned, accelerate to achieve their desired speed and then brake to the $25km/h$ following the leader vehicle, while IDM_3 keeps approaching to the previous vehicle (IDM_2).

During the third sequence (from $120s$ to $180s$), all vehicles are already close (in platooning) and all of them firstly accelerate to the desired value and then brake till the leader speed.

See now Figure 8d, focusing approximately to the $160s$ it is important to note that the braking force of the vehicles is in crescendo. Taking this in consideration, and with a certain amount of vehicles, it seems right to affirm that there would come a time that the braking force of one vehicle would be so high that this vehicle would even stop, creating a traffic jam. Thus, this simulation demonstrate that when there is a considerable amount of IDM vehicles following each other the string stability is in danger.

It is also important to note that the results in this simulation are more close to a real situation than the ACC vehicles simulation.

Finally, it is time now to try to check the impact of ACC autonomous cars introduced in ACC traffic flow.

7. Influence of autonomous driving in traffic

7.1. Simulation structure

The objective of this section is to develop a simulation platform for the interaction of the two models. You can see this model in detail in the current chapter.

This time, it is worth seeing in more detail the implementation of the model with *Matlab and Simulink*, named “MIX_ACC_IDM.m” and “simu_Control_MIX.slx” respectively.

First of all, it must be kept in mind that it is intended to simulate the behavior of the traffic flow in non-urban areas. In this scenario roads have no intersections, traffic lights, et cetera. Thus, the actions of the drivers are reduced. For these cases it is necessary to create a complex network of roads where vehicles do not stop circulating. In this case a circular circuit is chosen.

The construction of a circular model allows obtaining a constant flow of vehicles that do not stop circling around the circular path. It is also possible to keep an eye at the same cars all along the simulation, which helps to interpret results.

The parametric equations of the circuit model are:

$$(x, y) = (R \cdot \cos\theta\,,\ R \cdot \sin\theta) \quad \text{where} \quad \theta \,\epsilon\, [0,2\pi] \qquad \textit{(eq.23)}$$

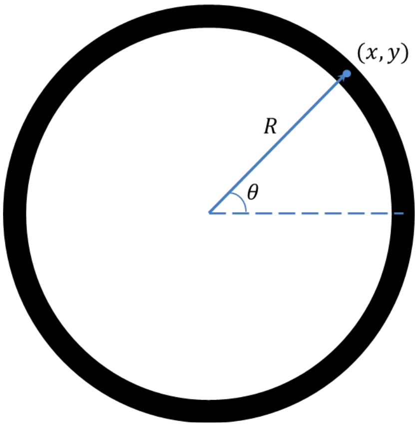


Figure 9: Circuit model

7.1.1. “MIX_ACC_IDM.m” MATLAB file

Focusing first on the .m file, it is structured as it follows:

1) Firstly the file “Functions_marc.m” is imported. It contains two functions which will be often used during the previous file. The functions are *generate* and *Coordinates2D*.
    a. The function *generate*, as its own name indicates, from a mean value, a standard deviation and the number of vehicles (N), generates a vector of length N with random values of the normal distribution. It will be used to assign random values to the parameters of the vehicles (thus, the simulation will be closer to reality).
    b. The function *Coordinates2D* returns the vertices of the rectangle that symbolizes the vehicle in the corresponding position of the vehicle in the circular circuit. This is done following these steps:
        - Using the length and width of the vehicle, the coordinates of the vertices of the rectangle with center at the origin $(0,0)$ are calculated (with the longitudinal axis of the rectangle parallel to the ordinate axis y).
        - The rectangle is oriented so with its longitudinal axis parallel to the tangent of the circuit at the point where the vehicle must be placed.
        - Finally, the oriented rectangle is displaced to the point where the vehicle should go, making it coincide with the center of the rectangle.

        For more details about the file “Functions_marc.m” see *ANNEX B.4.*

2) Then all the variables used during the simulation are set (circuit length, number of vehicles, which of them are autonomous, which position do this vehicles occupy, all the vehicles parameters, the control parameters, et cetera).
3) The simulation is executed using the Simulink file.
4) Finally, using the results obtained from the simulation, it is possible to construct some plots of the position on the road, the vehicles velocity and the distance between them. Also a graphic function of a previous work has been modified and used to be able to see the behavior of vehicles dynamically. It will be very useful to follow the vehicles and identify the behaviors of platooning and string stability.

For more detailed information about “MIX_ACC_IDM.m” see *ANNEX B.5*.

7.1.2. “simu_Control_MIX.slx” SIMULINK file

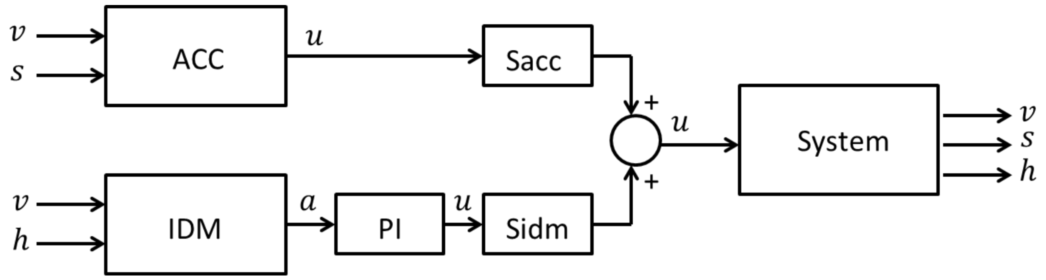


Figure 10: Scheme of the Simulink blocks in the mixed IDM and ACC simulations

This file contains the model of the simulation. It is principally composed by the following blocks:

- System: is in charge of enforcing the equations that characterize the dynamic model of study (Chapter 4.1 equations). See *ANNEX A.5*.
- ACC control: this control block characterizes the ACC control using Chapter 4.3 equations. See *ANNEX A.6*.
- IDM control: this control block characterizes the IDM control using Chapter 6.1 equations. From the velocity and the distance to the previous vehicle it calculates the necessary acceleration. See *ANNEX A.7*. As the input of the system is the necessary acceleration/braking force it is essential to implement a PI controller to find the corresponding force.
- PI controller : It is responsible to find the necessary acceleration/braking force from the acceleration given by the IDM control. Its constants (K_p, K_I) are calculated by pole assignment using an establishment time of $8s$ and a relatively small oscillation component $(w_p = 0{,}1\,rad/s)$. K_p and K_I become $(m, 0{,}3m)$ respectively. It is important to note that the velocity at the system is saturated at zero because in this model negatives velocities are not allowed. As this happens some physic errors are made at integral blocks and it its necessary to feedback the IDM block twice to amend the error committed (building a structure similar to an anti-windup control).
- Vectors Sacc and Sidm: there are vectors placed at IDM and ACC control outputs filled with zeros or ones depending of the type of control (Sacc filled with 1 if the vehicle is ACC and Sidm filled with 1 if the vehicle is IDM). The control outputs are multiplied by this vectors to assign the type of control regarding every vehicle and then added to enter the vector to the system block.

For more information about this model see *ANNEX A.4*.

7.2. Parameters used

During all simulations the number of vehicles is N=15 and the circuit length is $200\ m$. The vehicle and driver parameters are summarized in Table 4.

Vehicle parameter	***Mean***	***Deviation***	***Vehicle parameter***	***Value***	***Deviation***
Vehicle mass, m_i	$1000\ kg$	$100\ kg$	Minimum net distance, h_i^o	$2\ m$	$0{,}2\ m$
Gravity, g	$9{,}8\ m/s$	0	Safety time headway, T_{o_i}	$0{,}7\ s$	$0{,}2\ s$
Rolling coefficient, f_{r_i}	$0{,}0017$	$0{,}0003$	Desired (free road) speed,v_i^o	$60\ km/h$	$10\ km/h$
Air density, ρ	$1{,}225\ kg/m^3$	0	Maximum acceleration, a_i	$1\ m/s^2$	$0.2\ m/s^2$
Drag coefficient, C_{d_i}	$0{,}3$	$0{,}1$	Maxim deceleration, b_i	$3{,}5\ m/s^2$	$0{,}2\ m/s^2$
Vehicle frontal area, A_i	$2{,}8\ m^2$	$0{,}2\ m^2$	Acceleration exponent, δ_i	4	0
Road slope, $\propto$	0°	0	Initial speeds, v_i^o	$0\ km/h$	-
Vehicle length, l_i	$4\ m$	$4\ m$	Vehicle width, w_i	$1{,}9\ m$	$0{,}2$

Table 4: Parameters used during the mixed IDM and ACC simulation (extracted from [1] and [5]).

7.3. Simulation results

Using the structures, files and parameters commented, different simulations combining ACC and IDM vehicles are presented. Ten simulations of each case are made in order to calculate means and deviations to get more reliable results (since the initial values are randomly assigned).

The plots will show the position on road, velocity and distance to the previous vehicle (in gray for the IDM and in blue for the ACC).

Below there is an example of results for each case of study (all IDM, $1/3$ ACC, $2/3$ ACC and all ACC) and a summary of results where conclusions are drawn.

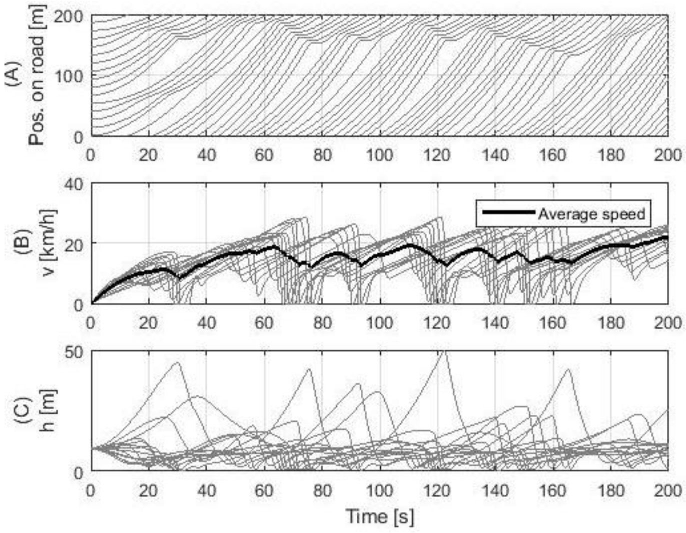


Figure 11: Simulation results: for all vehicles using IDM.

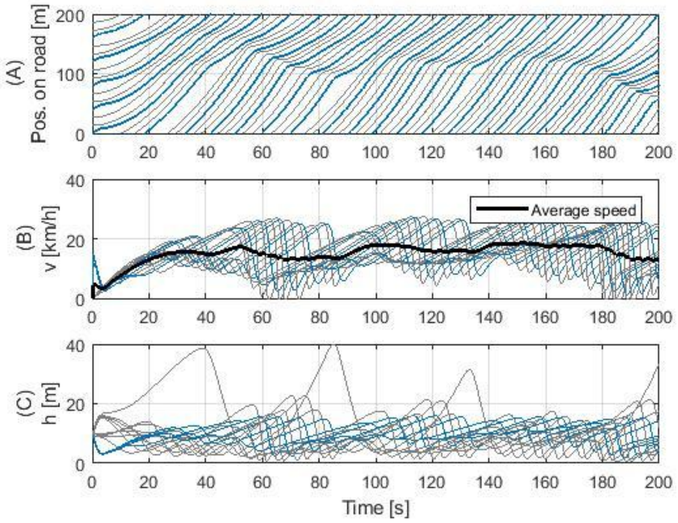


Figure 12: Simulation results: for 10 vehicles using IDM and 5 ACC.

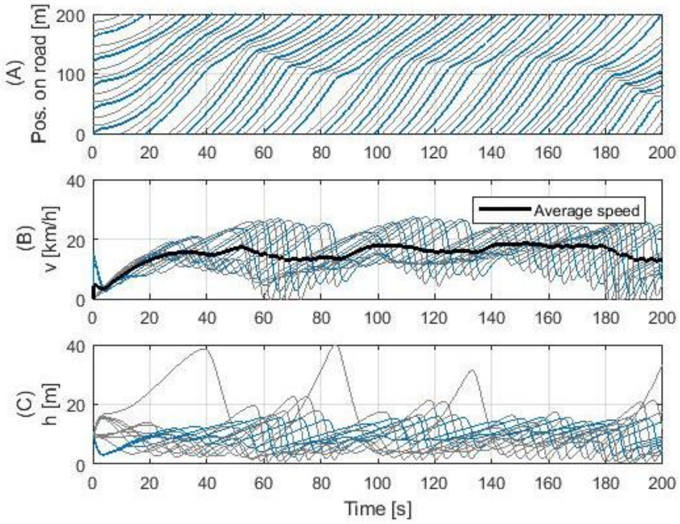


Figure 13: Simulation results: for 5 vehicles using IDM and 10 ACC.

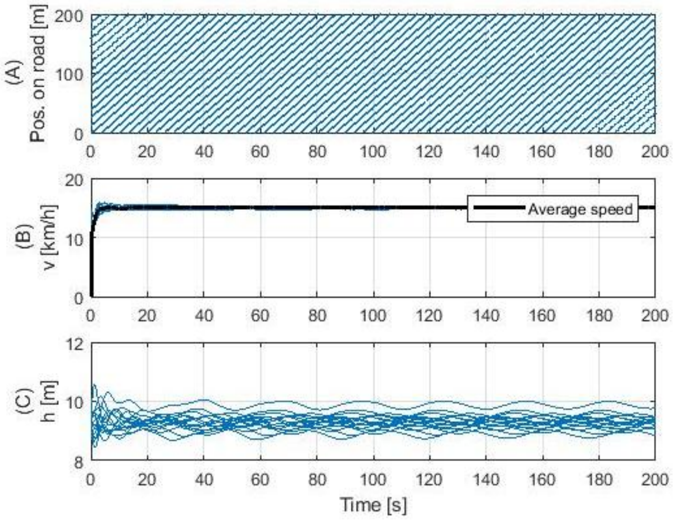


Figure 14: Simulation results: for all vehicles using ACC.

See at Table 5 the results from the different simulations mentioned above. It shows the average of average velocities and the average of the standard deviations.

Simulation case	***V average [km/h]***	***V standard deviation [km/h]***
15 IDM / 0 ACC	14,973	3,433
10 IDM / 5 ACC	13,996	2,840
5 IDM / 10 ACC	13,369	2,152
0 IDM / 15 ACC	15,094	0,429

Table 5: Summary of average results of each simulation case.

Notice that there are no significant changes in speed average, and also it is quite far from the desired speed. That it could be caused by a high density of cars with respect to the length of the track.

The most significant fact is the decrease of the standard deviation as the number of ACC vehicles increase. That means, introducing ACC vehicles (equipped with the controller of study) can help making the oscillations of the average speed much smaller, in fact almost zero. And this could mean an important improvement of traffic flow.

By decreasing the number of vehicles N or with a longer track possibly the average of velocities could grow and the improvement of the oscillations could be more significant but in this project it has been chosen to have a high density of vehicles to be able to notice the stop and go phenomenon. Anyway, in order to obtain the highest average speed possible, the minimum number of vehicles N has been chosen as the point where stop and go phenomenon begins to appear.

As a result, the model also generates a graphic function to be able to see the behavior of vehicles dynamically. See a screenshot of the resulting video at Figure 15.

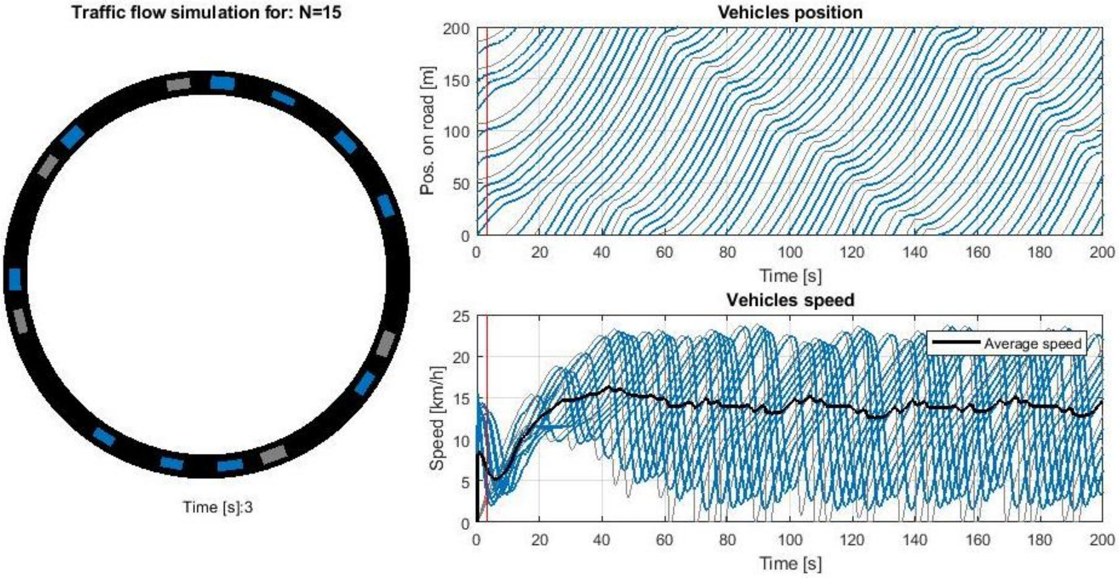


Figure 15: Example of the graphic function for N=15 (5 IDM and 10 ACC).

8. Budget

In the present chapter it is detailed an approximate total budget that carries the execution of the project, including both human and material resources.

In Table 6 it is indicated the different concepts and costs.

Concept	***Unit***	***Unitary Cost***	***Cost (€)***
Matlab and Simulink student software	1	69 €	69
Meetings	35 h	50 €/h	1750
Previous study	20 h	20 €/h	400
Programming	150 h	20 €/h	3000
Simulations	50 h	20 €/h	1000
COST			6219
IVA	21%		1306
Date: June 21st, 2018		***TOTAL COST***	***7525€***

Table 6: Budget of the project

The total approximated budget of the project, developed by a junior engineer, results 7525 €.

Conclusions

The most important challenge faced by this project has been to create a realistic simulation platform for the study of the interaction between autonomous and human-driven vehicles. This has been achieved by focusing on three key points.

The first one is to had studied and built an autonomous ACC driving model which allows switching between two different driving modes. This has provided very positive results, for example in terms of string stability.

The second one, the objective of reproducing real traffic situations has been met by introducing the behaviour of a driver with the IDM model.

The third one has been joining the previous ones in one larger model and test it in simulations.

Also, as has been shown in this project, simulation is a very powerful tool that allows us to study complex systems and predict their behaviour. Matlab and Simulink softwares have been ideal for this purpose.

There are also aspects which could be improved in the future as simulating varying the different parameters in the simulations as vehicle density, the autonomous vehicles order, the track length, et cetera. It can be also incorporated, for example, the change of lane, the exit and entry of cars and the simulation of accidents or emergency stops to make the simulation even more real.

Finally, the results obtained in this project prove that traffic flow would be improved by the incorporation of autonomous driven cars and it can even make the stop and go phenomena disappear. Thus, more research should be conducted on this promising technology to make it a reality in the near future.

Acknowledgements

First of all, I would like to thank the director of this project, Arnau Dòria Cerezo for the help and the guidance given during the project. His monitoring has been essential in order to finish the study.

I would also like to express my gratitude to my parents and friends, for all the support received during the last four years.

Bibliography

[1] A. D.-C. C. K. J. O. E. Benedito, «A variable structure-based algorithm for adaptive cruise control,» de *15th International Workshop on Variable Structure Systems and Sliding Mode Control (VSS).*, Graz (Austria), 9-11 July, 2018.

[2] P. F. A. S. S. M. di Bernardo, «Design, analysis, and experimental validation fo a distributed protocol for platooning int he presence of time-varying heterogeneous delays,» de *IEEE Trans. Intelligent Trnasportation Systems*, vol. 2, 2016, pp. 413-427.

[3] D.-S. N. v. d. W. a. H. N. J. Ploeg, «Controller sysnthesis for string stability of vehicle platoons,» de *IEEE Trans. Intelligent Transportation Systems*, vol. 2, 2014, pp. 854-865.

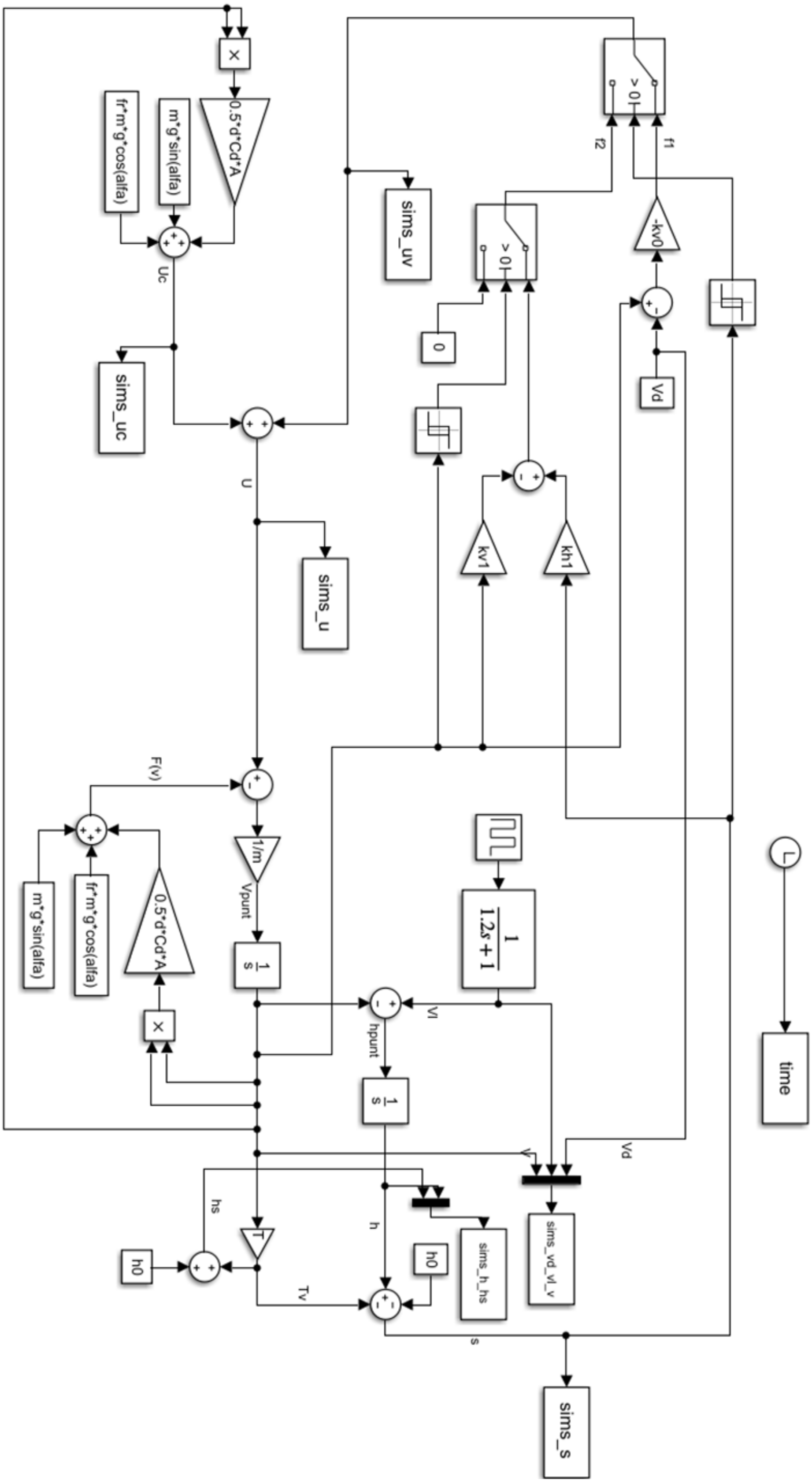
[4] M. Treiber, A. Hennecke i D. Helbing, «Congested traffic states in empirical observations and microscopic simulations,» 2000.

[5] A. D.-C. C. K. J. O. E. Benedito, «Traffic flow-oriented design and analysis of an adaptive cruise control system,» de *Proc. International Symposium on Circuits and Systems (ISCAS)*, Florence (Italy), May 2018.

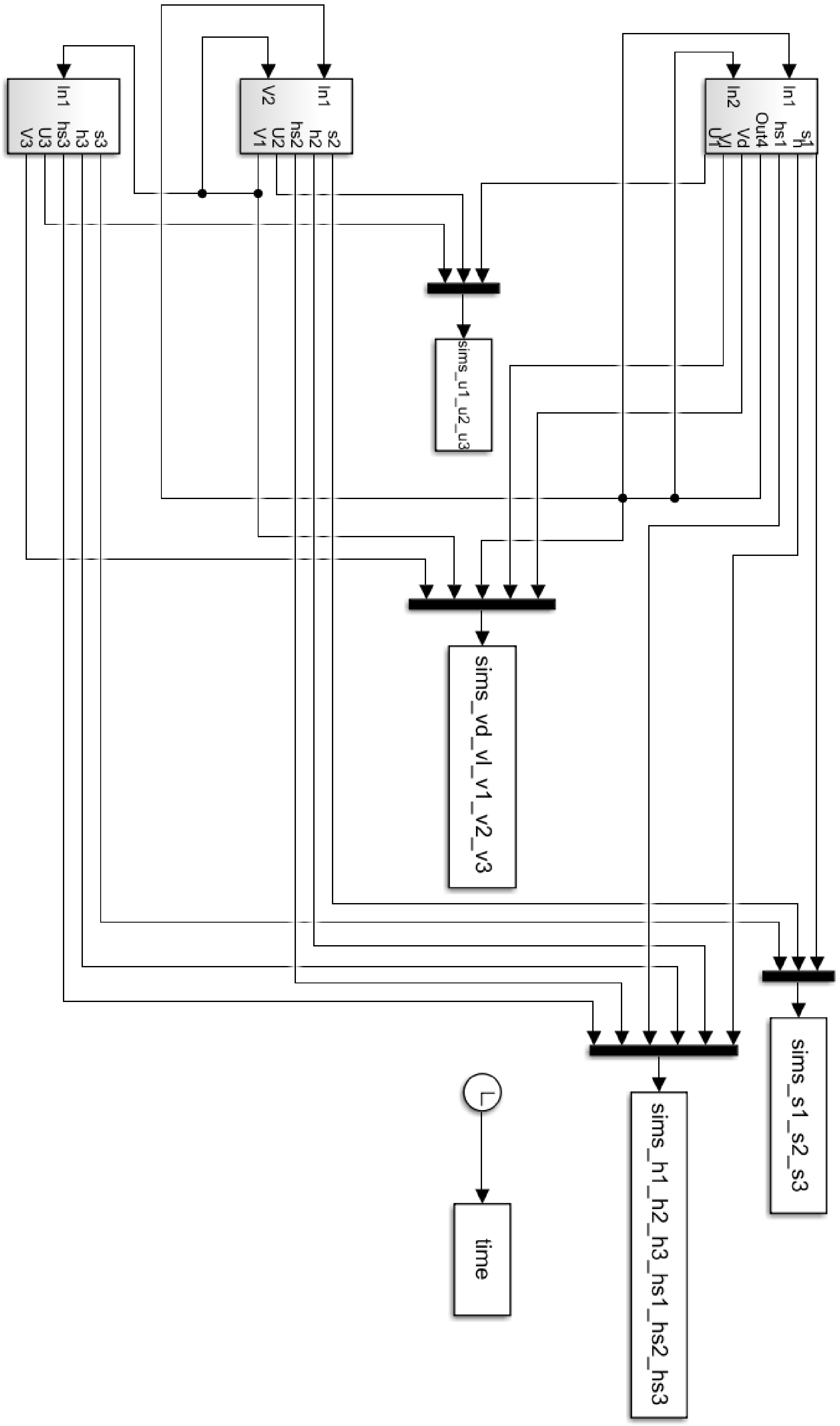
ANNEX

A. Simulink files

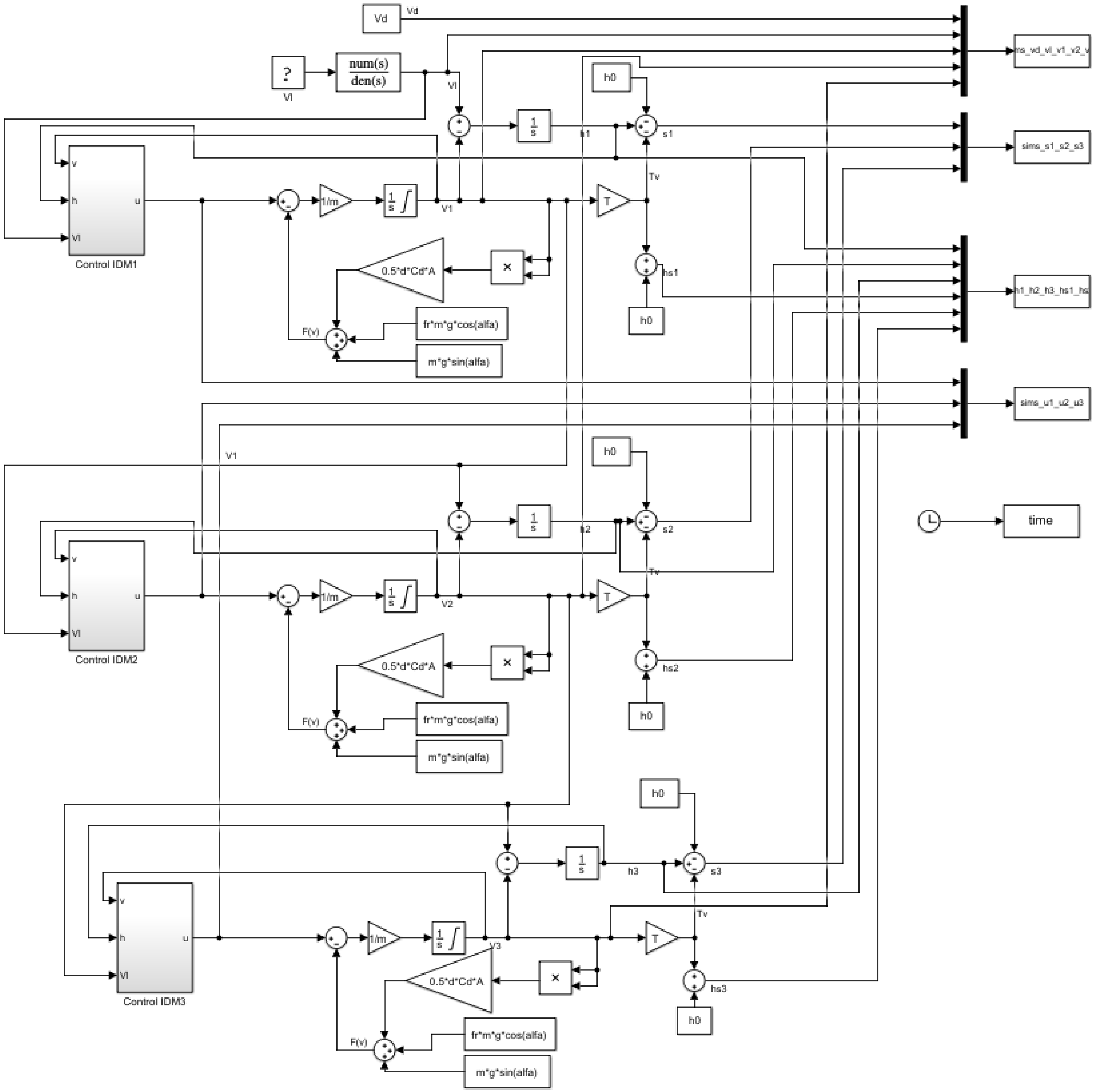
A.1 SCHEME FOR 1 ACC VEHICLE CONTROL



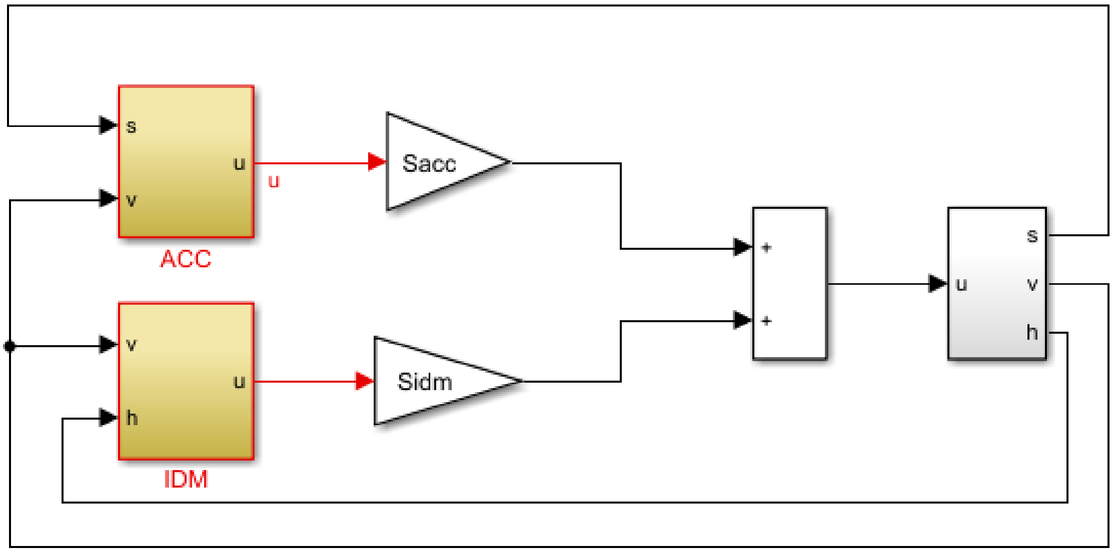
A.2 SCHEME FOR N ACC VEHICLES CONTROL (N = 3)



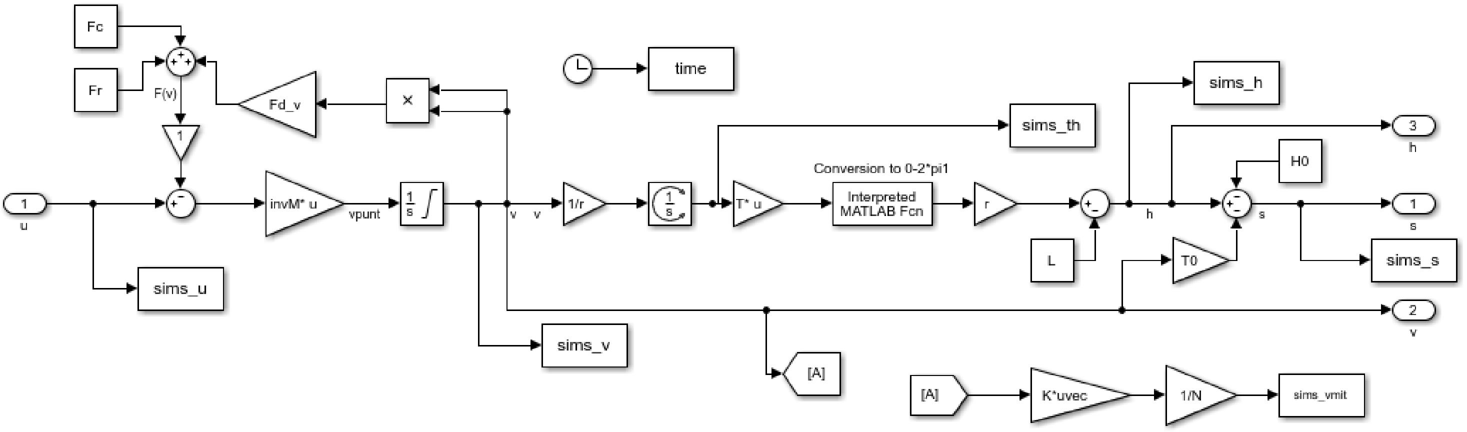
A.3 SCHEME FOR N IDM VEHICLES CONTROL (N = 3)



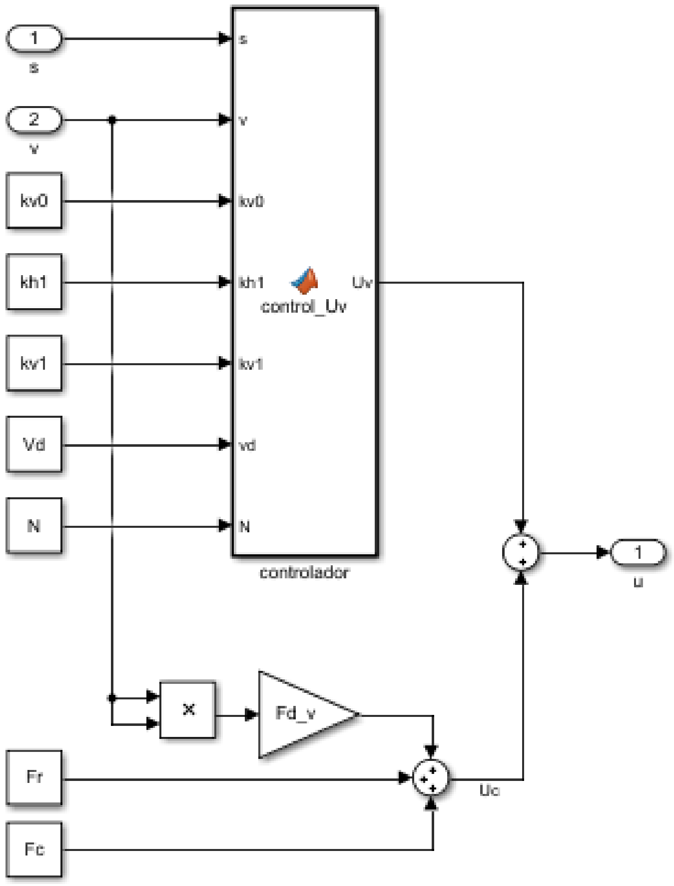
A.4 SCHEME FOR MIXED IDM/ACC CONTROL: GENERAL VIEW



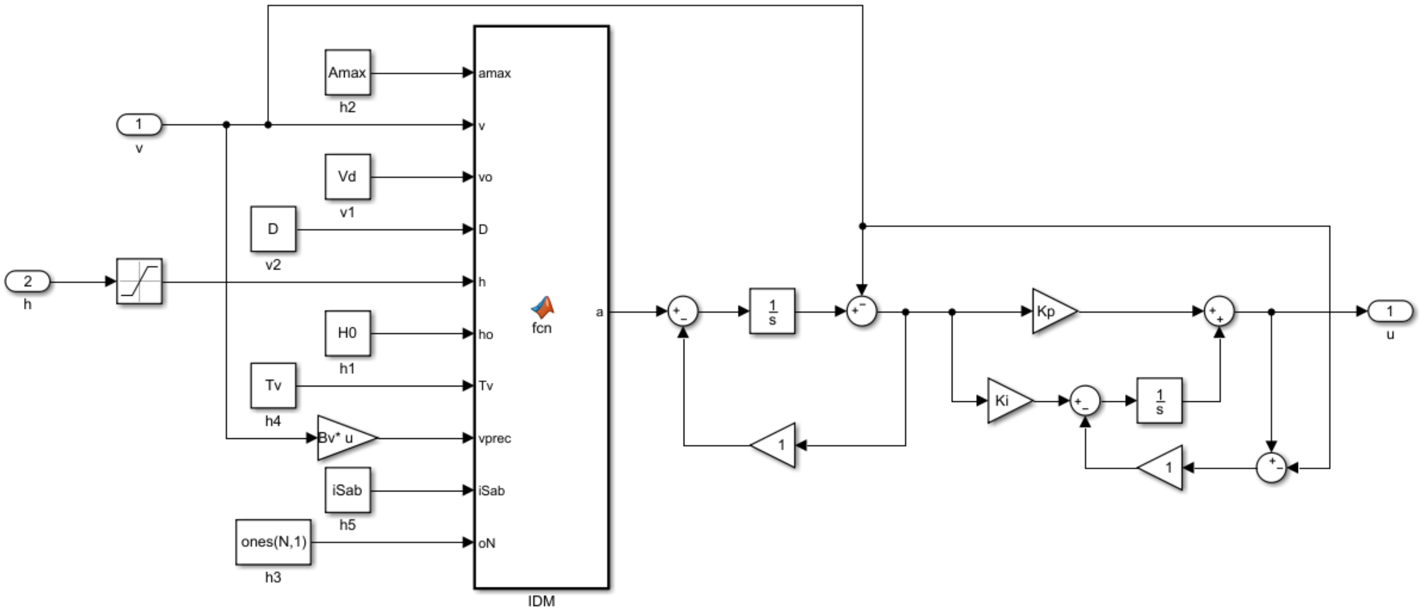
A.5 SCHEME FOR MIXED IDM/ACC CONTROL: SYSTEM BLOCK



A.6 SCHEME FOR MIXED IDM/ACC CONTROL: ACC CONTROL BLOCK



A.7 SCHEME FOR MIXED IDM/ACC CONTROL: IDM CONTROL BLOCK



B. Matlab files

B.1 PROGRAM FOR SIMULATION: LEADER AND ACC FOLLOWER

```
clear all;
close all;
clc

%Vehicle parameters:
m = 1000;           % vehicle mass
g = 9.8;            % gravity
alfa = 0;           % road slope
fr = 0.0017;        % rolling coefficient
d = 1.225;          % air density
Cd = 0.3;           % drag coefficient
A = 2.8;            % vehicle frontal area
h0 = 2;             % minimum desired distance
Vd = 70/3.6;        % desired speed
T = 1.7;            % time headway


%Control Parameters:
kv0 = 588;
kh1 = 600;
kv1 = 100;

%Initial conditions:
V = 0;              % initial follower speed
Vl = 0;             % initial leader speed
hin = 70;           % initial distance between vehicles
U0 = 0;             % initial follower acceleration force

%Simulation
tfinal = 80;                    % total simulation time
seq = [80 25 80 25]/3.6';       % velocity leader sequence
maxstepsize = 1e-3;             % maximum integration step
switchon = 1e-1;                % used in the saturation block
switchoff = -1e-1;              % used in the saturation block
sim('simu_Controlador1')        % simulation with simulink

% Plots
figure

subplot(4,1,1)           % h and hs plot
plot(time,sims_h_hs(:,1),'Linewidth',2)
hold on
plot(time,sims_h_hs(:,2),'k','Linewidth',2)
ylabel({'(A)';'h [m]'})
legend('h','h_s')
grid on

subplot(4,1,2)           % vd, vl and v plot
plot(time,sims_vd_vl_v(:,1)*3.6,'k','Linewidth',2)
hold on
plot(time,sims_vd_vl_v(:,2)*3.6,'r','Linewidth',2)
plot(time,sims_vd_vl_v(:,3)*3.6,'Linewidth',2)
```

```
ylabel({'(B)';'v [km/h]'})
legend('v_d','v_L','v')
grid on

subplot(4,1,3)          % s plot
plot(time,sims_s,'Linewidth',2)
ylabel({'(C)';'s [m]'})
grid on

subplot(4,1,4)          % u plot
plot(time,sims_u,'Linewidth',2)
hold on
ylabel({'(D)';'u [N]'})
grid on

xlabel('Time [s]')
```

B.2 PROGRAM FOR SIMULATION: LEADER AND N ACC FOLLOWERS (N = 3)

```
clear all;
close all;
clc

%Parameters:
m = 1000;
g = 9.8;
alfa = 0;
fr = 0.0017;
d = 1.225;
Cd = 0.3;
A = 2.8;
h0 = 2;
Vd = 70/3.6;
T = 1.7;

%Control Parameters:
kv0 = 588;
kh1 = 600;
kv1 = 100;

%Initial conditions:
V = 0;
Vl = 0;
hin = 70;
U0 = 0;

%Simulation
tfinal = 100;
seq = [80 25 25]/3.6';
maxstepsize = 1e-3;
switchon = 1e-1;
switchoff = -1e-1;
sim('simu_3ACC')

% Plots
```

```
close all
figure


subplot(4,1,1)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,1),'Linewidth',2)
hold on
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,2),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,3),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,4),'k','Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,5),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,6),'Linewidth',2)
ylabel({'(A)';'h [m]'})
legend('h1','h2','h3','h*1','h*2','h*3')
grid on

subplot(4,1,2)
plot(time,sims_vd_vl_v1_v2_v3(:,1)*3.6,'k','Linewidth',2)
hold on
plot(time,sims_vd_vl_v1_v2_v3(:,2)*3.6,'r','Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,3)*3.6,'Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,4)*3.6,'Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,5)*3.6,'Linewidth',2)
ylabel({'(B)';'v [km/h]'})
legend('v_d','v_L','v1','v2','v3')
grid on

subplot(4,1,3)
plot(time,sims_s1_s2_s3(:,1),'Linewidth',2)
hold on
plot(time,sims_s1_s2_s3(:,2),'Linewidth',2)
plot(time,sims_s1_s2_s3(:,3),'Linewidth',2)
ylabel({'(C)';'s [m]'})
legend('s1','s2','s3')
grid on

subplot(4,1,4)
plot(time,sims_u1_u2_u3(:,1),'Linewidth',2)
hold on
plot(time,sims_u1_u2_u3(:,2),'Linewidth',2)
plot(time,sims_u1_u2_u3(:,3),'Linewidth',2)
ylabel({'(D)';'u [N]'})
legend('u1','u2','u3')
grid on

xlabel('Time [s]')
```

B.3 PROGRAM FOR SIMULATION: LEADER AND N IDM FOLLOWERS (N = 3)

```
clear all;
close all;
clc

%Vehicle parameters:
m = 1000;       % vehicle mass
g = 9.8;        % gravity
alfa = 0;       % road slope
fr = 0.0017;    % rolling coefficient
d = 1.225;      % air density
Cd = 0.3;       % drag coefficient
A = 2.8;        % vehicle frontal area
h0 = 2;         % minimum desired distance
Vd = 60/3.6;    % desired speed
T = 1.7;        % time headway

%IDM
amax=1;
bmax = 3.5;
delta=4;
Ts=0.7;
iSab=1/(2*sqrt(amax*bmax));

%PI
Kp = 50*m;
Ki = 0.3*m;

%Initial conditions:
Vini = 0;          % initial follower speed
Vl = 0;         % initial leader speed
hini1 = 70;       % initial distance between leader and vehicle 1
hini2 = 150;
hini3 = 230;
U0 = 0;         % initial follower acceleration force

%Simulation
tfinal = 180;                 % total simulation time
seq = [80 25 25]/3.6';    % velocity leader sequence
maxstepsize = 1e-3;          % maximum integration step
sim('simu_Control_NIDM')     % simulation with simulink

% Plots
figure

subplot(4,1,1)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,1),'Linewidth',2)
hold on
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,2),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,3),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,4),'k','Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,5),'Linewidth',2)
plot(time,sims_h1_h2_h3_hs1_hs2_hs3(:,6),'Linewidth',2)
ylabel({'(A)';'h [m]'})
legend('h1','h2','h3','h*1','h*2','h*3')
grid on
```

```
subplot(4,1,2)
plot(time,sims_vd_vl_v1_v2_v3(:,1)*3.6,'k','Linewidth',2)
hold on
plot(time,sims_vd_vl_v1_v2_v3(:,2)*3.6,'r','Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,3)*3.6,'Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,4)*3.6,'Linewidth',2)
plot(time,sims_vd_vl_v1_v2_v3(:,5)*3.6,'Linewidth',2)
ylabel({'(B)';'v [km/h]'})
legend('v_d','v_L','v1','v2','v3')
grid on

subplot(4,1,3)
plot(time,sims_s1_s2_s3(:,1),'Linewidth',2)
hold on
plot(time,sims_s1_s2_s3(:,2),'Linewidth',2)
plot(time,sims_s1_s2_s3(:,3),'Linewidth',2)
ylabel({'(C)';'e [m]'})
legend('e1','e2','e3')
grid on

subplot(4,1,4)
plot(time,sims_u1_u2_u3(:,1),'Linewidth',2)
hold on
plot(time,sims_u1_u2_u3(:,2),'Linewidth',2)
plot(time,sims_u1_u2_u3(:,3),'Linewidth',2)
ylabel({'(D)';'u [N]'})
legend('u1','u2','u3')
grid on

xlabel('Time [s]')
```

B.4 PROGRAM FOR SIMULATION: MIXED IDM AND ACC (Functions_marc)

```
function f = Functions
f.genera = @generate;
f.Coord2D = @Coordinates2D;

end

    function [y]=generate(mit,desv,N)

        for i=1:N
            y(i)=normrnd(mit,desv);
        end
    end

    function [Xk,Yk] = Coordinates2D(h,l,theta,r)

        %Vertex at (0,0)
        V = [-h,l;h,l;h,-l;-h,-l].*0.5;
        M_rotactio = [cos(-theta),sin(-theta);-sin(-theta),cos(-theta)];
        %Vertex appliying orientation
        Vf = (M_rotactio*V')';
        %Moving vertexs to curve
```

```
        Vf2 = Vf +
r.*[cos(theta),sin(theta);cos(theta),sin(theta);cos(theta),sin(theta);cos
(theta),sin(theta)];
        Xk = Vf2(:,1);
        Yk = Vf2(:,2);
    end
```

B.5 PROGRAM FOR SIMULATION: MIXED IDM AND ACC

```
clear all; close all; clc

f=Functions_marc; %Functions that will be used during the program

%Parameters:
C=200;          % Circuit lenght(m)
r=C/2/pi;       % Radius
N=15;           % Number of vehicles
%Nmax = C/(1+h0);
acc = [2,4,5,6,8,10,12,13,14,15];     % ACC IDM mix
%acc=[];                   %all IDM
%acc=linspace(1, N, N); %all ACC

g = 9.81;
d = 1.225;   %Air density
T0 = 1.7;

Cd0 = 0.3;   %Drag coefficient
Cd = f.genera(Cd0,0.1,N);

A0 = 2.8;    %Frontal Area
A = f.genera(A0,0.2,N);

fr0 = 0.0017;  %Friction coefficient
frs = 0.0003;
fr = f.genera(fr0,frs,N);

alfa0 = 0*(2*pi/360);   %slope
alfa = f.genera(alfa0,0,N);

vd0=80;    % Mean freeroad speed (in km/h)
vds=10;
Vd=f.genera(vd0,vds,N)/3.6;

W0 = 1.9;  % Vehicle Width
Ws = 0.2;
W = f.genera(W0,Ws,N);

L0 = 4;      %Vehicle lenght
Ls = 0.2;
L = f.genera(L0,Ls,N);

m0 = 1000;   %Vehicle mass
ms = 100;
m = f.genera(m0,ms,N);
```

```
%%%%%%%%%%Càlculs%%%%%%%%%%

%%%%%IDM%%%%%
amax=1;
amaxs=0.2;
Amax = f.genera(amax,amaxs,N);

bmax = 3.5;
bmaxs = 0.2;
Bmax=f.genera(bmax,bmaxs,N);

for i=1:N
    iSab(i)=1/(2*sqrt(Amax(i)*Bmax(i)));
end

delta=4;
deltas=0;
D = f.genera(delta,deltas,N);

h0=2;
h0s=0.2;
H0 = f.genera(h0,h0s,N);

tv=0.7;
tvs=0.2;
Tv = f.genera(tv,tvs,N);

for i=1:N
   if i==1
       Bv(i,N)=1;
   else
       Bv(i,i-1)=1;
   end
end

%%%%%Controlador PI per conversió a IDM%%%%%
%PI with poles at -0.5+-j0.1 (ts=8s and wp=0.1) --> Kp=m ,Ki = 0.3.*m
Kp = m;
Ki = 0.3.*m;

%%%%%ACC%%%%%
invm = 1./m;
M = diag(m);
invM = inv(M);

t1=diag(-1*ones(1,N));
t2=diag(ones(1,N-1),-1);
t3= zeros(1,N);
t3(1,N)=1;
t1(1,:)=t1(1,:)+t3;
T = t1+t2;

Fr = fr.*m*g.*cos(alfa);
Fc = m*g.*sin(alfa);
Fd_v = 0.5*d.*Cd.*A;

%%%%%%%%%%Control parameters%%%%%%%%%%
kv0o = 588;
```

```
kv0 = f.genera(kv0o,0,N);

kh1o = 600;
kh1 = f.genera(kh1o,0,N);

kv1o = 100;
kv1 = f.genera(kv1o,0,N);

%%%%%%%%%%Type of control%%%%%%%%%%
Sidm=ones(1,N);
Sacc=zeros(1,N);

for i=1:length(acc)
   aux=acc(i);
   Sidm(aux)=0;
   Sacc(aux)=1;
end

%%%%%%%%%%Initial conditions%%%%%%%%%%
v0 = zeros(1,N);

for i=1:N
    th0(i)=2*pi*(N-i)/N;
end

u0 = zeros(1,N);

%%%%%%%%%%Simulation%%%%%%%%%%
tfinal = 200;
maxstepsize = 1e-3;
sim('simu_Control_MIX')

%%%%%%%%%%Plots%%%%%%%%%%
close all

%Colors
Color_idm=[1 1 1]*0.5;
Color_acc=[0 0.4470 0.7410];

subplot(3,1,3) %h
for i=1:N
    if Sacc(i)==1
        plot(time,sims_h(:,i),'Color',Color_acc,'Linewidth',1)
    elseif Sidm(i)==1
        plot(time,sims_h(:,i),'Color',Color_idm,'Linewidth',1)
    else
        plot(time,sims_h(:,i),'r','Linewidth',1)
    end
    hold on
end
ylabel({'(C)';'h [m]'})
grid on
xlabel('Time [s]')

subplot(3,1,2) %v
for i=1:N
    if Sacc(i)==1
        plot(time,sims_v(:,i)*3.6,'Color',Color_acc,'Linewidth',1)
```

```
    elseif Sidm(i)==1
        plot(time,sims_v(:,i)*3.6,'Color',Color_idm,'Linewidth',1)
    else
        plot(time,sims_v(:,i)*3.6,'r','Linewidth',1)
    end
    hold on
end
av = plot(time,sims_vmit*3.6,'-k','Linewidth',2);
legend(av, 'Average speed');
ylabel({'(B)';'v [km/h]'})
grid on

subplot(3,1,1) %u
for i=1:N
    if Sacc(i)==1
        col=Color_acc;
        lw=1.5;
    else
        col=Color_idm;
        lw=1;
    end
    [aux,loc]=findpeaks(sims_th(:,i));
    ind2=loc(1)-1;
    plot(time(1:ind2),sims_th(1:ind2,i)*r,'Color',col,'Linewidth',lw)
    hold on
    for j=1:length(loc)-1
        ind1=loc(j)+1;
        ind2=loc(j+1)-1;

plot(time(ind1:ind2),sims_th(ind1:ind2,i)*r,'Color',col,'Linewidth',lw)
    end
    ind1=loc(end)+1;
    plot(time(ind1:end),sims_th(ind1:end,i)*r,'Color',col,'Linewidth',lw)
end

ylabel({'(A)';'Pos. on road [m]'})
grid on
%%

%%%%%% ANIMATION %%%%%%

vid = VideoWriter('sims','MPEG-4');
vid.FrameRate = 100;
myVideo.Quality = 500;

theta = linspace(0,2*pi);

figure(2)
set(figure(2), 'Position', [0 0 1200 500]);
hold on;
subplot(2,7,[1 3 8 10],'Color','k')
format long;
Text = text(-4,-1.2*r,strcat('Time [s]: '));
r1=r-2;r2=r+2;xf=0;Xf=0;yf=0;Yf=0;

x = xf + r1*cos(theta);
y = yf + r1*sin(theta);
X = Xf + r2*cos(theta);
```

```
Y = Yf + r2*sin(theta);

patch([x X],[y Y],'k','linestyle','non');
title(['Traffic flow simulation for: N=' num2str(N)])
axis([-1.2*r 1.2*r -1.2*r 1.2*r]);
axis off;hold on;axis equal;

%Inizializating cars
X_acc = [];X_idm = [];
Y_acc = [];Y_idm = [];

for i = 1:N
     [Xi,Yi] = f.Coord2D(W(i),L(i),sims_th(1,i),r);

      if Sacc(i)==1
          X_acc = [X_acc,Xi];
          Y_acc = [Y_acc,Yi];

      else
          X_idm = [X_idm,Xi];
          Y_idm = [Y_idm,Yi];
      end
end
p1= patch(X_acc,Y_acc,Color_acc);
p2 = patch(X_idm,Y_idm,Color_idm);

% Position graphic
figure(2)
subplot(2,7,[4 7])
h = animatedline;
for i=1:N
    if Sacc(i)==1
        col=Color_acc;
        lw=1.5;
    else
        col=Color_idm;
        lw=1;
    end
    [aux,loc]=findpeaks(sims_th(:,i));
    ind2=loc(1)-1;
    plot(time(1:ind2),sims_th(1:ind2,i)*r,'Color',col,'Linewidth',lw)
    hold on
    for j=1:length(loc)-1
        ind1=loc(j)+1;
        ind2=loc(j+1)-1;

plot(time(ind1:ind2),sims_th(ind1:ind2,i)*r,'Color',col,'Linewidth',lw)
    end
    ind1=loc(end)+1;
    plot(time(ind1:end),sims_th(ind1:end,i)*r,'Color',col,'Linewidth',lw)
end

xlabel('Time [s]');
ylabel('Pos. on road [m]')
title('Vehicles position');

t1 = line([0 0],ylim,'Color','r');
grid on;hold on;
```

```
% Speed graphic
figure(2)
subplot(2,7,[11 14])
sims_vav_in=sims_v(:,1)*0;
sims_vav_out=sims_v(:,1)*0;
nout=0;
nin=0;
for i=1:N
    if  Sacc(i)==1
        col=Color_acc;
        lw=1.5;
    else
        col=Color_idm;
        lw=1;
    end
    plot(time,sims_v(:,i)*3.6,'Color',col,'Linewidth',lw)
    hold on
end

av = plot(time,sims_vmit*3.6,'-k','Linewidth',2);
ylabel('Speed [km/h]')
grid on
t2 = line([0 0],ylim,'Color','r');
grid on;hold on;
legend(av, 'Average speed');
title('Vehicles speed');
xlabel('Time [s]');
ylabel('Speed [km/h]');

% Update simulation
Counter = 0;
At_video = 60;
for i=1:At_video:length(time)
     X_acc = [];X_idm = [];
     Y_acc = [];Y_idm = [];
    for k = 1:N
        [Xi,Yi] = f.Coord2D(W(k),L(k),sims_th(i,k),r);
        if Sacc(k)==1
          X_acc = [X_acc,Xi];
          Y_acc = [Y_acc,Yi];
        else
          X_idm = [X_idm,Xi];
          Y_idm = [Y_idm,Yi];
        end
    end

    set(p1,'XData',X_acc,'YData',Y_acc);
    set(p2,'XData',X_idm,'YData',Y_idm);
    set(t1,'XData',[time(i) time(i)]);
    set(t2,'XData',[time(i) time(i)]);
    set(Text,'String',strcat('Time [s]: ',num2str(time(i),'% 3.0f')));

    open(vid);
    V = getframe(gcf);
    writeVideo(vid,V);
end
close(vid);
```