



**Politecnico
di Torino**

Corso di Laurea Magistrale in
Ingegneria Meccanica

Tesi di Laurea Magistrale

**Development of an Adaptive Model Predictive Control for
platooning safety in Battery Electric Vehicles**

Relatori:

Prof.ssa Daniela Anna Misul
Dott. Ing. Alessia Musa
Dott. Ing. Matteo Spano

Tutor Aziendale:

Ing. Gianluca Toscano

Candidato:

Antonio Capuano

A.A. 2020/2021

Table of Contents

Abstract	3
1. Introduction	4
2. Model Predictive Control	6
3. BEV Model in Simulink	12
3.1 Parameters of the BEV model	22
4. Changes in the BEV model	24
5. MPC Toolbox	34
5.1 Cooperative Adaptive Cruise Control (CACC) [19]	34
5.2 MPC Parameters	35
5.2.1 MPC structure and internal plant model [20]	35
5.2.2 Sample time, prediction horizon and control horizon [22]	36
5.2.3 Scale factors [23]	38
5.2.4 Constraints [24]	39
5.2.5 Weights [25]	40
5.2.6 Cost function [26]	42
5.2.7 Robustness and aggressiveness	44
5.3 Adaptive Cruise Control block [27].....	44
5.4 Custom MPC	49
6. Adaptive Model Predictive Control [28]	53
6.1 Validation of the controller on different driving cycles	57
6.1.1 FTP72	58
6.1.2 WLTP	61
6.1.3 US06	63

6.1.4 Random cycle	66
6.1.5 Results analysis	67
7. Four vehicle platoon simulation	70
8. Conclusions and future works	79
Figures Index.....	80
Table Index.....	83
Bibliography.....	84

Abstract

Nowadays, the continuous improvement in transportation systems technologies provides several different opportunities which are exploited for the enhancement of safety and comfort in passenger vehicles. As an example, Adaptive Cruise Control (ACC) might provide benefits, including smoothness of the traffic flow and collision avoidance. In addition, Vehicle-to-Vehicle (V2V) communication may be exploited in the car-following model to obtain further improvements in safety and comfort by guaranteeing fast response to critical events. In this work, firstly an Adaptive Model Predictive Control is developed for managing the Cooperative ACC scenario of two vehicles; as a second step, the safety analysis during a cut-in manoeuvre is performed, extending the platooning vehicles number to four. The effectiveness of the proposed methodology is proved in different driving scenarios such as diverse cruising speeds, steep accelerations and aggressive decelerations. Moreover, the controller is validated by considering various speed profiles of the leader vehicle, including a real drive cycle obtained using a random drive cycle generator software. Results show that the proposed control strategy is capable of quickly responding to unexpected manoeuvres and of avoiding collisions between the platooning vehicles, still ensuring a minimum safety distance in the considered driving scenarios.

1. Introduction

The growth of the car fleet on the road leads to an increase of traffic accidents, environmental pollution and oil shortage. To solve these problems advanced driver assistance systems (ADAS) and battery electric vehicles (BEVs) are two important tools. Indeed, integrating these two advanced technologies, vehicles can achieve better driving safety, convenience and environmental friendliness.

ADAS, in these prospective, covers a major role to improve drive safety and comfort. Therefore, they are applied amongst a multitude of situations to assist the driver avoiding forward collisions, keeping the lane, braking automatically and guaranteeing pedestrian safety. The first generation of safety applications was designed using local sensors such as cameras and radars. Then, other sources of information are employed to provide more accurate information. Between them particularly relevant is vehicle-to-vehicle (V2V) communication, that allows to send information about the state of the vehicle to other vehicles without considering a specifically topological position. Indeed, its omnidirectional connectivity capabilities permits to adopt different topological communication structure to improve performance. This technology covers a main role into the development of new features as Cooperative Adaptive Cruise Control (CACC).

Cooperative Adaptive Cruise Control is the evolution of Adaptive Cruise Control that take advantage of V2V communication to acquire information from the surrounding vehicles and drive the vehicle simultaneously avoiding collisions and maximizing traffic throughput. Using this tool, the ego vehicle receives information about the speed, position and acceleration of the surrounding vehicles and defines the control actions to safely follow the preceding vehicle and generating a platoon. To achieve the optimization for multiple objectives of CACC system, a proper control algorithm is needed. There are many control algorithm for solving multi-objective optimization, for example dynamic programming or genetic algorithm, however Model Predictive Control (MPC) is one of the most effective methods, because, using its receding prediction horizon, can find optimal solution online while compensating errors due to inaccurate modelling.

Therefore, the purpose of this work is to develop a MPC controller in order to equip a BEV with CACC, enhancing its safety, drivability and comfort. To correctly fulfil this goal, different

phases have to be completed: at first a bibliographical analysis was carried to collect information about different types of Model Predictive Control (e.g. Adaptive MPC, Nonlinear MPC, etc...) and choose the most appropriate one for the case study. Then, the model of the battery electric vehicle was developed in MATLAB/Simulink using the parameters of a real passenger car, namely 500e. Afterwards, the design of the controller was defined with the aid of the MPC Toolbox of MATLAB/Simulink. The so obtained controller was validated on several drive cycles demonstration the effectiveness of the algorithm on safety and comfort. At the end, to test the reactivity of the controller, an application involving a four vehicle platoon and a cut-in manoeuvre was simulated.

2. Model Predictive Control

First of all is important to proceed in analysing Model Predictive Control properties.

Model Predictive Control is one of the most successful techniques adopted in industry to control multivariable systems in an optimized way under constraints on input and output variables. In MPC, the manipulated inputs are computed in real time by solving a mathematical programming problem, most frequently a Quadratic Program (QP). [1] The QP depends on a model of the dynamics of the system, that is often learned from experimental data. To adopt MPC in embedded control systems under fast sampling and limited CPU and memory resources, one must be able to solve QP's with high throughput, using simple code and executing arithmetic operations under limited machine precision, and to provide tight estimates of execution time.

The MPC uses a model of the plant to make predictions on the future plant output behaviour and, also, employs an optimizer which ensures that the predicted future plant outputs track the desired reference (Figure 1).

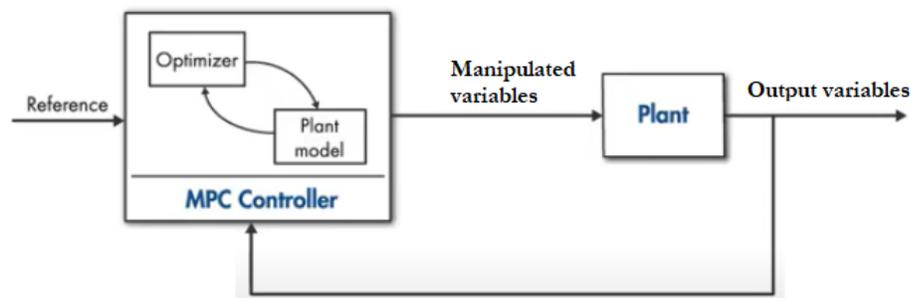


Figure 1 Model Predictive Control design

At the current time step the MPC uses the plant model to simulate the plant paths in the next time steps (Figure 2). How far ahead the MPC controller can look into the future is called prediction horizon. The MPC controller needs to find the best prediction path that is closest to the reference.

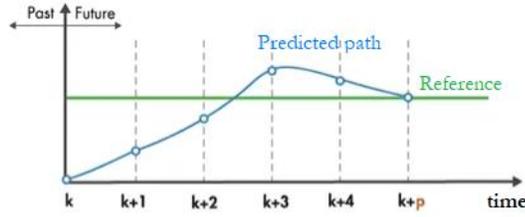


Figure 2 MPC possible solution

To do so, multiple future scenarios are simulated in a systematic order. Indeed, the optimizer, by solving an online optimization problem, tries to minimize the error between the reference and the predicted path. It also tries to minimize the change in the actuator signal from one step to another [2]. All these features can be summarized in the cost function J that has to be minimized and is a trade-off between tracking the reference signal and reducing the variation of the actuator signal or put them near to an optimal level. An example of a general cost function is the following one:

$$J = \sum_{i=1}^p w_e e_{k+i}^2 + \sum_{i=0}^{p-1} w_{\Delta u} \Delta u_{k+i}^2 \quad (1)$$

where:

- p is the preceding horizon;
- e and Δu are the error from the reference and the variation in the actuator signal at time step $k + i$, respectively;
- w_e and $w_{\Delta u}$ are the tuneable weights of each part of the cost function.

While minimizing this cost function, the MPC makes sure that the output of the system and the actuator signals stay within predetermined limits, referred as constraints.

At the current time step, the MPC controller solves the optimization problem over the prediction horizon several times while satisfying the constraints. The predicted solution with the lowest J is the optimal one and, therefore, that control sequence is selected. Of the latter, only the first step is applied to the plant and the rest of the sequence is disregarded.

At the next time step, the real new states of the plant are the input of the MPC and, shifting the prediction horizon from one time step, the controller repeats the same cycle of calculations to compute the optimum control sequence for the next time step. Using the information of the states of the vehicle and comparing with the values predicted from the MPC, also disturbance can be taken into account.

In the MPC framework two important parameters are the sample time and the control horizon. Choosing proper values for these parameters is important as the effect not only influence the controller performance but also the computation complexity of the MPC algorithm.

The sample time determines the rate at which the controller executes the control algorithm. If it is too big the presence of a disturbance won't allow the controller to react in proper time. On the contrary, if the sample time is too small, the controller can react much faster to disturbances but it can cause an excessive computational load.

Instead, the control horizon is another important parameter. It represents the number of control moves to the time step m that lead to the predicted future output, as shown in Figure 3.

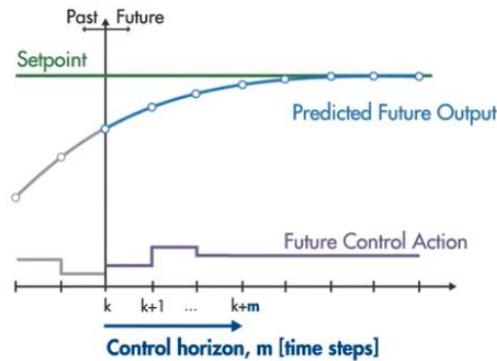


Figure 3 Example of control horizon

The rest of control actions, over the prediction horizon, are held constant. Each control selected in the control horizon can be thought as a free variable that needs to be computed by the optimizer and the smaller is the control horizon, the fewer are the computations. However, if a control horizon of only one step is chosen, it might not give the best possible path. On the contrary, increasing the control horizon can give better predictions at the cost of increasing the complexity. At most the control horizon can be chosen the same length as the prediction

horizon, but only the first control moves have a significant effect on the predicted output behaviour, therefore it only increases computational complexity [3].

The MPC models are distinguished in linear and nonlinear model predictive control: the linear MPC relies on a linear system, linear constraints and a quadratic cost function, therefore the link between the inputs and outputs is linear. A MPC problem, that has these properties, gives rise to a convex optimization problem where the cost function has a single global optimum and the goal of the optimization is to find this optimum. Instead, if the system plant is nonlinear, it is possible to linearize it.

If the system is nonlinear, as the constraints and/or the cost function, a nonlinear MPC is needed. This method uses the most accurate representation of the plant but it is the most challenging one to solve in real time because the optimization problem is non-convex and the cost function may have many local optima. Therefore, finding the global optimum may be hard and the efficiency of solving the non-convex optimization problem relies on the nonlinear solver [4].

Another important kind of MPC is the Adaptive MPC. With this control algorithm it is possible to deal with nonlinear systems and still benefit from the nice properties of the convex optimization problem. Indeed, in Adaptive MPC a linear model, that approximates the nonlinear system well near the operating point, is computed as the operating conditions change and at each time step, you update the internal plant model used by the MPC controller with this linear model [4].

In conclusion the MPC is an extremely flexible control design approach that owns these advantages [1]:

- Prediction model can be multivariable, with delays, disturbances, etc;
- It can exploit available preview on future references and measured disturbances;
- It handles constraints on inputs and outputs.

Nevertheless there are also price to pay:

- It requires a model that has to be built before starting the optimizations;
- There are many degrees of freedom, which increase the complexity by the tuning point of view;

- It requires real-time computations to solve the optimization problem.

The MPC has been involved in a lot of applications such as process control, aerospace, aeronautic, automotive, etc...

In particular, it fits very well in automotive applications for its real-time capability.

In this work, several paper have been analysed, in which the MPC is used in combination with Advanced Driver Assistance Systems (ADAS) to obtain the best results in terms of energy economy, tracking capability or robustness of the prediction. This analysis aims to understand how to model the vehicle system, implement the cost function with its constraints and which are the possible results.

Specifically, papers [5, 6, 7, 8] consider “conventional” NMPCs in which a vehicle model and a cost function are considered, and starting from this model, minimizes a certain objective. Furthermore, three of these texts consider vehicles equipped with V2V which allows them to have a CACC (or eCACC) which allows to integrate different information from the fleet of vehicles to obtain an optimized speed profile and consequently reduced energy consumption. The only exception is “*Robust Model Predictive Cooperative Adaptive Cruise Control Subject to V2V Impairments*” which shows an MPC in which the previous vehicle, through the V2V, sends to the following vehicle a sequence of possible future acceleration commands, used to reduce errors due to the possible loss of information packets. In this paper was shown the capability of Modern Predictive Control to guarantee a robust control and handle possible disturbances in the application considered. Papers [9, 10, 11], on the other hand, present MPCs with particular characteristics: an SNMPC, namely a NMPC that considers stochastic input parameters useful for improving control, in this case the possible speed profile of the preceding vehicle estimated at each iteration [10], a MPC that integrates offline a dynamic programming (DP), which calculates the optimal speed profile and saves it in a cloud to be used online by the MPC, and an IDP-MPC, which is a system that defines the problem through MPC and solves it with an iterative dynamic programming. Also in these papers the communication between vehicles and GPS information are exploited for the operation of ADAS technologies. Using the combination of this controller the performances of MPC can be improved and move closer to that of offline solvers, as DP.

Finally, in [12] was evaluated the effect on the health of the battery that is obtained from the implementation of an optimal speed control system, taking into account the limits related to

regenerative braking. In the described work, only 50% of the vehicles are connected, so a traffic prediction will be required for the other vehicles. Despite this, the result obtained in terms of energy consumption is not much lower than that obtained with all connected vehicles.

Based on this analysis, the proposal for the development of the work would be to develop a vehicle model which includes longitudinal dynamics, engine and battery dynamics and then design a MPC to implement a CACC, through the use of V2V communication, in order to improve safety and consumption. Furthermore, some “atypical” MPC functions could be considered to improve the overall performance of the model (for example, considering stochastic parameters as input) and allow the simulation of more complex driving scenarios.

3. BEV Model in Simulink

To build properly a MPC framework an essential part is to model a plant of the system that you want to control and optimize. This can be one of the main parts of the work, maybe the longer one, because without a proper plant model of the system, the simulation will never lead to the desired solutions.

In this case study, the longitudinal model of a battery electric vehicle is obtained in Simulink. This model is ruled by the several equations of its components: the battery, DC motor, vehicle body, tyres, ecc...

The vehicle body equations represents a two axle vehicle in its longitudinal motion. The vehicle motion is a result of the net effect of all the forces and torques acting on it.

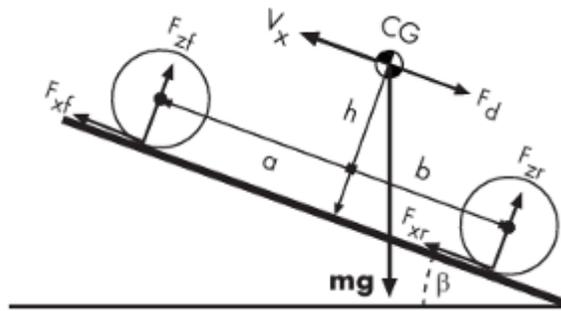


Figure 4 Longitudinal dynamics of the vehicle

The equations are the following:

$$m\dot{V}_x = n(F_{xf} + F_{xr}) + \frac{1}{2}C_d\rho A(V_x + V_w)^2 \cdot \text{sgn}(V_x + V_w) - mg \cdot \sin\beta \quad (2)$$

where:

- m is the mass of the vehicle;
- V_x is the longitudinal velocity of the vehicle;
- n is the number of wheels on each axle;
- F_{xf}, F_{xr} are the longitudinal forces on each wheel at the front and rear ground contact points, respectively;

- C_d, ρ, A are the aerodynamic drag coefficient, mass density of air and effective frontal vehicle cross-sectional area, respectively;
- V_w is the wind speed;
- g is the gravitational acceleration;
- β is the grade angle.

Also normal forces on the wheels are considered:

$$\begin{aligned}
 F_{zf} &= \frac{-h \left(\frac{1}{2} C_d \rho A (V_x + V_w)^2 \cdot \text{sgn}(V_x + V_w) + mg \cdot \sin\beta + m\dot{V}_x \right) + b \cdot mg \cos\beta}{n(a + b)} \\
 F_{zr} &= \frac{+h \left(\frac{1}{2} C_d \rho A (V_x + V_w)^2 \cdot \text{sgn}(V_x + V_w) + mg \cdot \sin\beta + m\dot{V}_x \right) + a \cdot mg \cos\beta}{n(a + b)}
 \end{aligned} \tag{3}$$

where:

- h is the height of vehicle centre of gravity (CG) above the ground;
- The wheel normal forces satisfy: $F_{zf} + F_{zr} = mg \frac{\cos\beta}{n}$;
- a, b are the distance of front and rear axles, respectively, from the normal projection point of vehicle CG onto the common axle plane.

The tyres longitudinal behaviour are represented with an empirical equation based on four fitting coefficients, called Pacejka Magic Formula. The structure of the formula is the following one:

$$y = D \cdot \sin \left[C \cdot \arctg \left(Bx - E \left(Bx - \arctg(Bx - \arctg(Bx)) \right) \right) \right] \tag{4}$$

where:

- x and y are the input and output of the equation, respectively;
- B is the stiffness factor;
- C is the shape factor;
- D is the peak factor;

- E is the curvature factor.

The Tyre Magic Formula assumes longitudinal motion only and includes no camber, turning, or lateral motion. Also, the coefficient are calibrated by experimental tests.

The DC motor is represented by the equivalent circuit model shown in Figure 5.

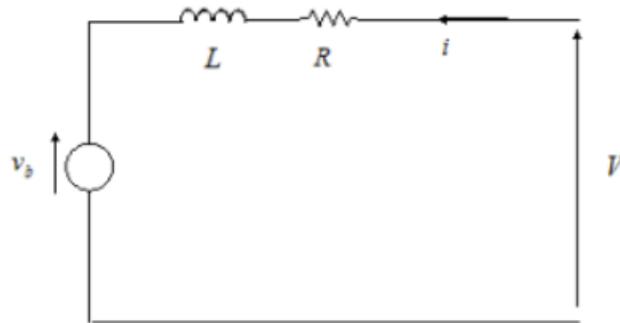


Figure 5 Equivalent circuit model

The equation that describes this model is the following one:

$$V(t) = Ri(t) + L \frac{di}{dt} + v_b(t) \quad (5)$$

where:

- R and L are the resistance linked to the dispersions of the electrical circuit and the inductance of its winding, respectively;
- i and V represent, respectively, the armature current and voltage affecting the rotor windings;
- v_b is the back emf induced in the armature by the permanent magnets.

The back emf is given by the equation:

$$v_b(t) = k_v \omega(t) \quad (6)$$

with k_v the back emf constant and ω the motor angular speed.

The description of the mechanical part of the motor is given by the Newton second law:

$$J_m \frac{d\omega(t)}{dt} = T(t) - T_c(t) - \lambda\omega \quad (7)$$

where:

- J_m represents the motor inertia;
- $T(t)$ is the torque supplied by the motor due to the effect of the induced Lorentz force and it is given by $T(t) = k_t i(t)$ with k_t the torque constant;
- $T_c(t)$ is the load torque;
- λ is the motor damping coefficient.

In this model it is assumed that there are no electromagnetic losses. This means that the mechanical power is equal to the electrical power dissipated by the back emf in the armature and equating these two terms gives:

$$\begin{aligned} T\omega &= v_b i \\ k_t &= k_v \end{aligned} \quad (8)$$

namely the torque constant is equal to the back emf constant.

The DC motor gets the energy from the battery. The most simple battery equivalent model is the internal resistance model and it is represented in Figure 6.

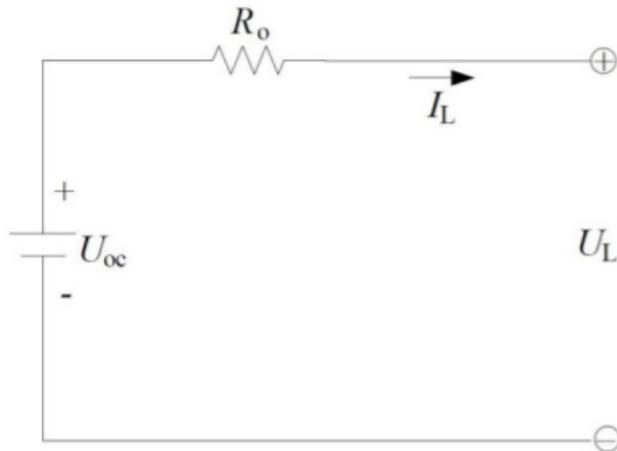


Figure 6 Internal resistance model

The equation that rules this simple model is the following one:

$$U_L = U_{OC} - R_0 I_L \quad (9)$$

where:

- U_{OC} is the open circuit voltage;
- R_0 is the internal resistance;
- I_L is the battery current;
- U_L is the load voltage.

These three elements are tied together among themselves and allow the power transfer necessary to the vehicle's movement.

A complete BEV model is shown in Figure 7.

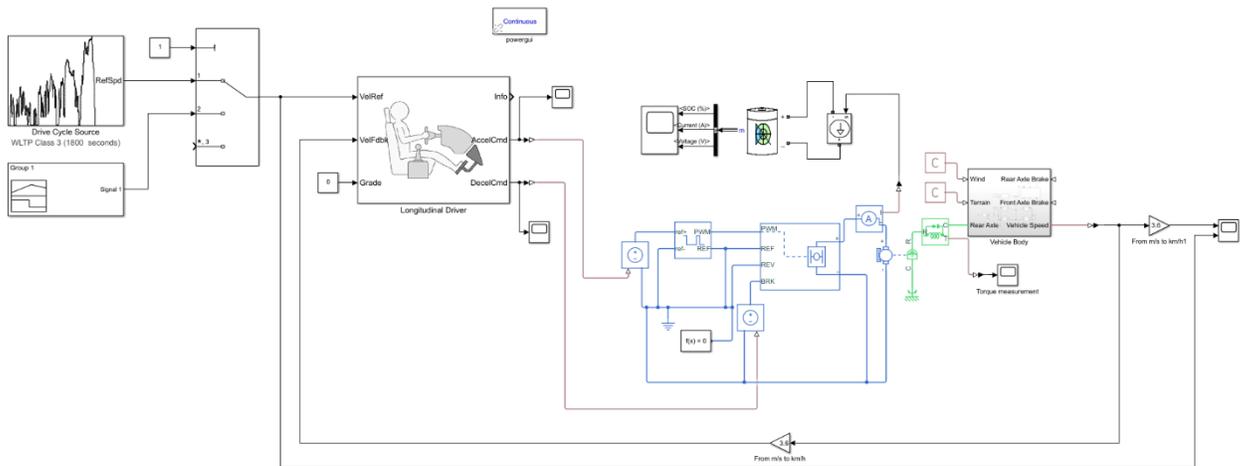


Figure 7 BEV model

To build this model in Simulink some important steps are needed. The first elements that must be modelled are the tyres and the vehicle body and can be grouped in a subsystem called, in the developed model, vehicle body. Figure 8 exhibit the elements of this subsystem.

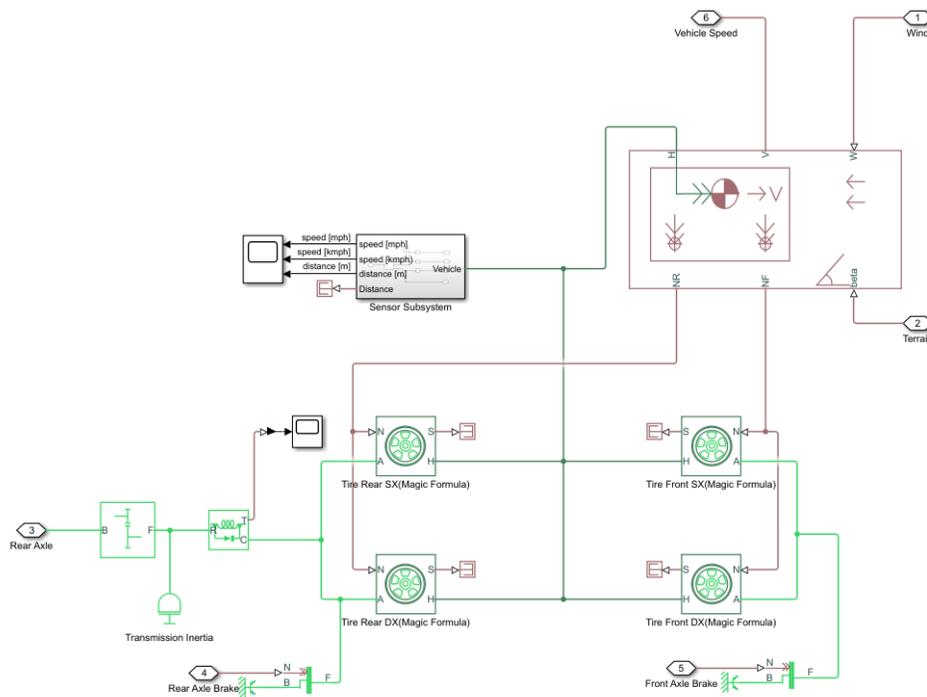


Figure 8 Vehicle body subsystem

In this subsystem a Tyre (Magic Formula) block is used to simulate the tyres behaviour. This block models the tyre longitudinal behaviour through the Pacejka magic formula. The constant coefficients used in the magic formula are displayed in Table 1 for different road conditions [13].

Surface	B	C	D	E
Dry tarmac	10	1.9	1	0.97
Wet tarmac	12	2.3	0.82	1
Snow	5	2	0.3	1
Ice	4	2	0.1	1

Table 1 Magic Formula Coefficients for Typical Road Conditions

The block takes in input the normal force on the tyre and the gives the slip factor. All the tyres of a same axle are connected between axle port to have the same rotational motion and the hub ports of the 4 tyres are linked to the vehicle body to give traction motion to the vehicle. Also the friction brakes for the two axles and the traction torque are connected to the axle port.

Instead, the vehicle body block represents a two-axle vehicle body in longitudinal motion. The block accounts for body mass, aerodynamic drag, road incline, and weight distribution between axles due to acceleration and road profile. The rolling resistance is considered into tyre block. The outputs of the block are the vehicle speed and the normal forces on the two axles. The equations used in these blocks are those described previously.

To complete the subsystem, a simple gear block is used to transmit the output torque of the motor. Since the simple gear do not consider the transmission inertia, an inertia block is needed.

The input torque to the subsystem is provided by the DC motor block. The DC Motor block represents the electrical and torque characteristics of a DC motor using the equivalent circuit model in Figure 5 and specifying its circuit parameters. Port C represent the DC motor case and it is connected to a rotational reference. This motor is controlled by the Controlled PWM Voltage and H-Bridge blocks: the first one represents a pulse-width modulated voltage source, the second is the motor driver that allows inversion of current and direction of rotation to enable regenerative braking.

A PWM signal is a square wave with a variable “duty cycle” (the ratio between the time in which the square wave takes on a “high” value and the period T), that can control the power drained by the electric load varying the “duty cycle”. The demanded duty cycle is determined by the block through the following expression:

$$100 * \frac{V_{ref} - V_{min}}{V_{max} - V_{min}} [\%] \quad (10)$$

where:

- V_{ref} is the reference voltage across the ref+ and ref- ports;
- V_{min} is the minimum reference voltage;
- V_{max} is the maximum reference voltage.

Then the block sends the signal at the output voltage amplitude to the H-bridge. The H-Bridge is also needed because a PWM motor driver goes open circuit in between pulses.

The H-Bridge is a simple electrical circuit, that is composed by four electronic switches which control a load placed in the centre and it is represented in Figure 9.

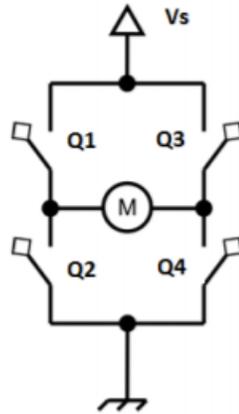


Figure 9 H-Bridge scheme

It allows to drive the load only in two modes: ON, closing two switches on the opposite sides of the bridge (e.g. Q1 and Q4 or Q3 and Q2), and OFF, opening all the switches or creating an open circuit. When the state of the bridge is ON the motor is powered and runs at the maximum speed allowed by the voltage V_s . On the contrary, when the state is OFF the motor is not powered. Combining the H-Bridge with the PWM control a continuous adjustment of the motor rotation speed can be obtained. In the H-Bridge block selecting the Averaged and Smoothed mode the output sent to the load is:

$$\frac{V_o V_{PWM}}{A_{PWM}} - I_{out} R_{ON} \quad (11)$$

where:

- V_o is the value of the output voltage amplitude parameter;
- V_{PWM} is the value of the voltage at the PWM port;
- A_{PWM} is the value of the PWM signal amplitude parameter;
- I_{out} is the value of the output current;
- R_{ON} is the bridge on resistance parameter.

To speed up the simulation both the Controlled PWM Voltage and the H-Bridge blocks have to be set on Averaged mode.

The output current of the H-bridge block is read by the current sensor and sent to a controlled current source. Thus, it can be connected with the vehicle battery and simulate its process of discharge/charge. The battery is described by using a battery block that represents most popular types of rechargeable batteries. Figure 10 shows the equivalent circuit that the block models.

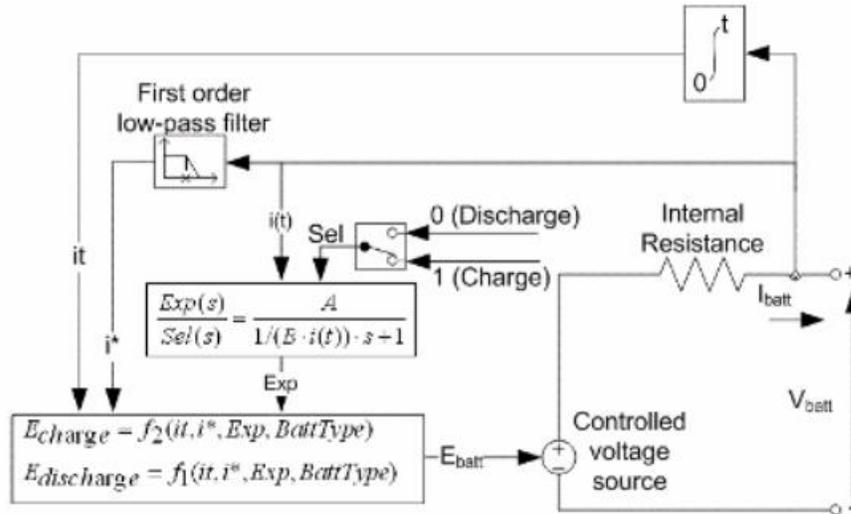


Figure 10 Battery model block

Considering the lithium-ion battery type, the equations used for the charge and discharge models are:

$$\begin{aligned}
 E_{discharge} &= E_0 - K \cdot \frac{Q}{Q - it} \cdot i^* - K \cdot \frac{Q}{Q - it} \cdot it + A \cdot \exp(-B \cdot it) \\
 E_{charge} &= E_0 - K \cdot \frac{Q}{it + 0,1 \cdot Q} \cdot i^* - K \cdot \frac{Q}{Q - it} \cdot it + A \cdot \exp(-B \cdot it)
 \end{aligned} \tag{12}$$

where:

- E_0 is the constant voltage;
- K is the polarization constant;
- i^* is the low-frequency current dynamics;

- Q is the maximum battery capacity;
- it is the extracted capacity;
- A and B are the exponential voltage and capacity, respectively.

The block give an output vector of signal for the battery current, voltage and SOC (if desired, also battery age, temperature, maximum capacity and ambient temperature can be displayed). In particular the state of charge is calculated as:

$$SOC = 100 \left(1 - \frac{1}{Q} \int_0^t i(t) dt \right) \quad (13)$$

To command the whole system and convert the speed data from a drive cycle (the drive cycle source block is used) or from a pre-set speed profile into an acceleration and braking command, a controller is required. The selected one is a driver that implements a longitudinal speed driver controller. Based on reference and feedback velocities, the Longitudinal Driver block implements a Proportional-integral (PI) control with tracking anti-windup and feed-forward gains. To calculate the speed control output, the block uses this equation:

$$y = \frac{K_{ff}}{v_{nom}} v_{ref} + \frac{K_p e_{ref}}{v_{nom}} + \int \left(\frac{K_i e_{ref}}{v_{nom}} + K_{aw} e_{out} \right) dt + K_g \theta \quad (14)$$

where:

- $e_{ref} = v_{ref} - v$;

- $e_{out} = y_{sat} - y$;

- $y_{sat} = \begin{cases} -1 & y < -1 \\ y & -1 \leq y \leq 1; \\ 1 & 1 < y \end{cases}$

- v_{nom}, v_{ref}, v are the nominal vehicle speed, the reference velocity signal and the feedback velocity signal, respectively;

- e_{ref} and e_{out} are the velocity error and the difference between saturated (y_{sat}) and nominal control outputs (y), respectively;

- K_{ff}, K_{aw} are the velocity feed-forward gain and anti-windup gain, respectively;
- K_p and K_i are proportional and integral gain, respectively;
- K_g is the grade angle feed-forward gain;
- θ is the grade angle.

In this way is possible to generate the commands necessary to track a longitudinal drive cycle, delivered by different source such as Drive Cycle Source block or an excel file imported in Simulink.

3.1 Parameters of the BEV model

The vehicle model represents a real passenger car, namely Fiat 500e. To properly reproduce the motion of this vehicle several parameters or maps are required:

- Vehicle mass m ;
- Horizontal distance from CG (centre of gravity) to front and rear axle;
- CG height above the ground;
- Frontal area of the vehicle;
- Drag coefficient;
- Wheel radius;
- Rolling resistance coefficient;
- Gear ratio and inertia of the gearbox;
- DC armature resistance and inductance;
- DC back-emf constant;
- DC rotor inertia and damping;
- Battery voltage, capacity and initial SOC;
- H-bridge resistance and freewheeling diode resistance;
- PWM frequency;
- Maximum braking force.

Unfortunately, not all of these parameters is available in literature, therefore some of them can be replaced with maps. The available parameters are [14]:

Vehicle mass	1474 kg	Gear ratio	9,59
Wheel radius	0,278 m	Initial SOC	0,8
Drag coefficient	0,311	Front Track	1,4 m
Frontal area	2,1 m ²	Rear Track	1,4 m

Table 2 Fiat 500e parameters

4. Changes in the BEV model

To reduce the simulation time and incorporate the available data and maps for the simulated vehicle, some changes are needed in the BEV model.

The resulting model is shown in Figure 11.

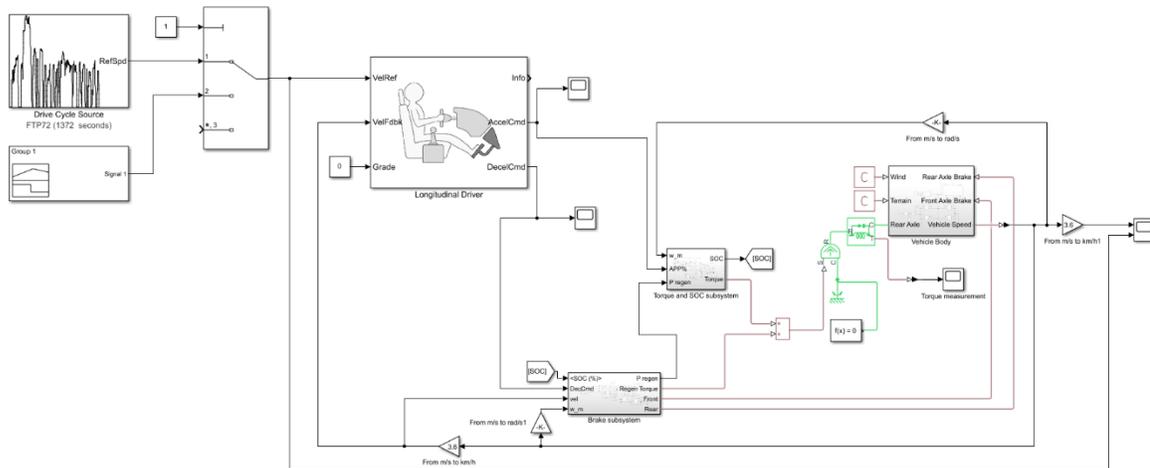


Figure 11 BEV model after changes

The first important difference consists in eliminating the PWM control and H-Bridge blocks that increase remarkably the simulation time. These blocks have been replaced by two subsystems called “Torque and SOC subsystem” and “Brake subsystem” as seen in Figure 11. In these subsystems, the required torque is determined using the DC motor map and this torque is sent to the vehicle thanks to the ideal torque source. Indeed, at the current vehicle speed, the maximum motor torque or brake torque is obtained from the motor map and it is multiplied to the normalized acceleration and deceleration command provided by the controller.

The map used is that of an electric motor with the maximum mechanical power of 87 kW and shown in Figure 12. The map has been scaled using a design optimization tool developed in Politecnico di Torino [15, 16].

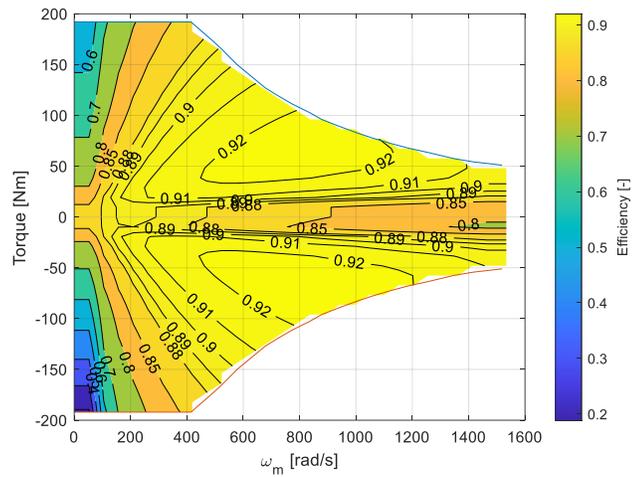


Figure 12 Efficiency map of the motor

In Figure 12, the contour for the same efficiency are represented with one colour as shown by the colour bar.

The part of the map where the torque is positive is implemented in a lookup table in the Torque and SOC subsystem. Instead, the negative part is included in the Brake subsystem. The two subsystems are exhibited in Figure 13 and Figure 14, respectively.

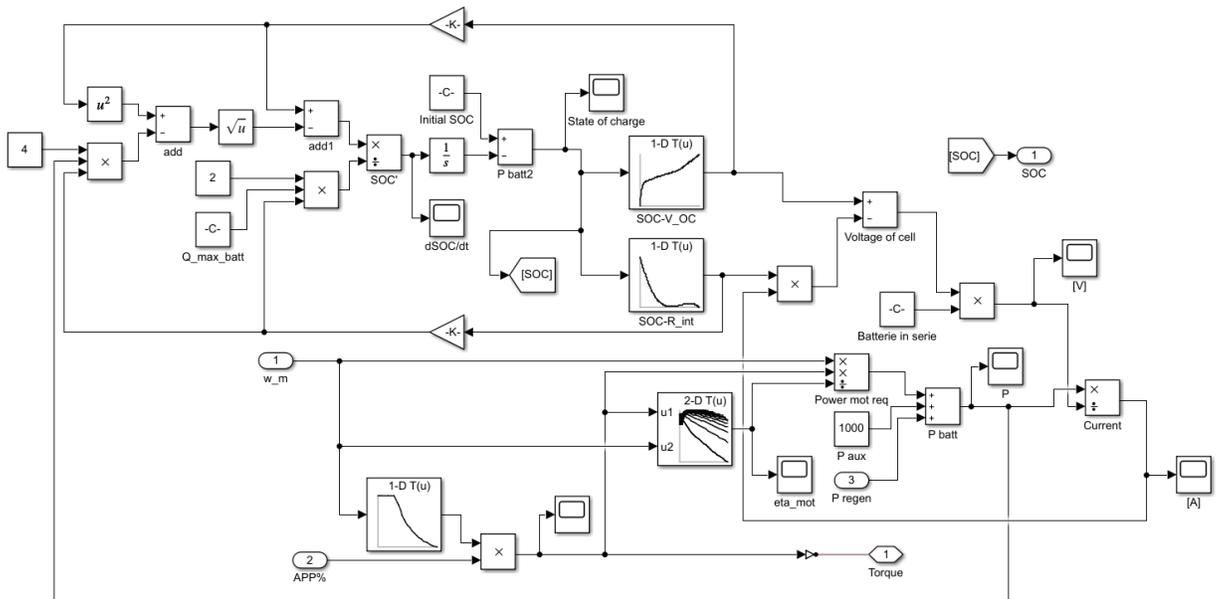


Figure 13 Torque and SOC subsystem

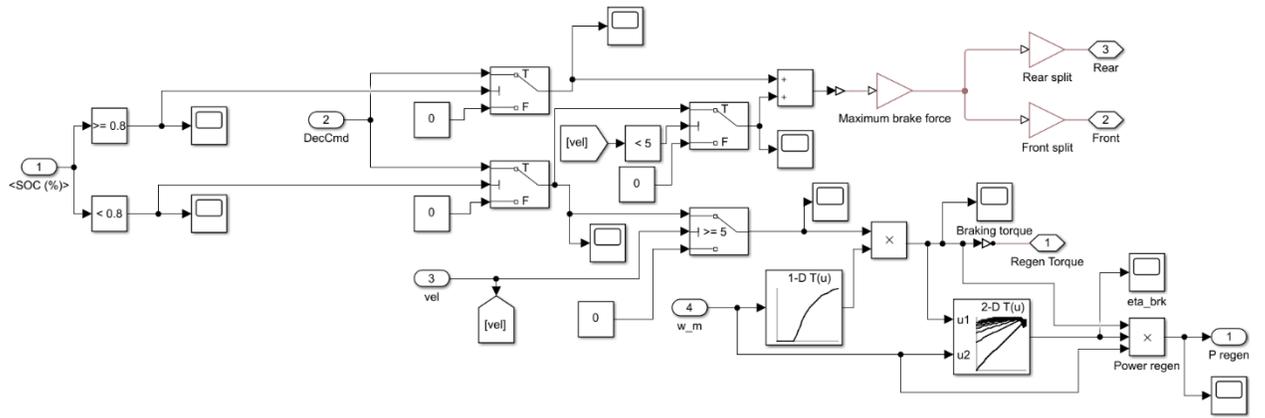


Figure 14 Brake subsystem

In both the subsystems, depending on the angular motor speed, it is possible to obtain the maximum torque at that speed from a lookup table and this value is multiplied by the acceleration or deceleration command. This torque is sent to the ideal torque source to move or brake the vehicle and it is also used to determine the motor efficiency in that operating point and compute the required power or the power generated by the motor. In particular, for the brake subsystem this only happens if the vehicle speed is higher than 5 km/h , because below this threshold the regenerative braking efficiency is too low, so the frictional braking is preferred. Also, if the state of charge of the battery is lower than 80% , regenerative braking is disabled to avoid overcharging of the battery and damaging its performance.

The required or generated power by the motor is added to the power required from the auxiliaries, conventionally set to 1000 W . This gives the power required or provided to the battery.

Another difference between the previous model and the current one is how the state of charge is estimated. In fact, with the generic battery block the SOC was determined through an integral of the battery current. Instead, in the subsystem of Figure 13, it is possible to calculate the rate of change of the SOC with the following equation:

$$\dot{SOC} = - \frac{V_{OC} - \sqrt{V_{OC}^2 - 4P_{batt}R_{int}}}{2R_{int}Q_{batt}} \quad (15)$$

where:

- V_{OC} is the open circuit voltage;
- P_{batt} and Q_{batt} are the battery power and the battery capacity, respectively;
- R_{int} is the interna resistance of the battery.

In this model the open circuit voltage and the battery internal resistance depend on the instantaneous battery state of charge, as shown in Figure 15 [17, 18].

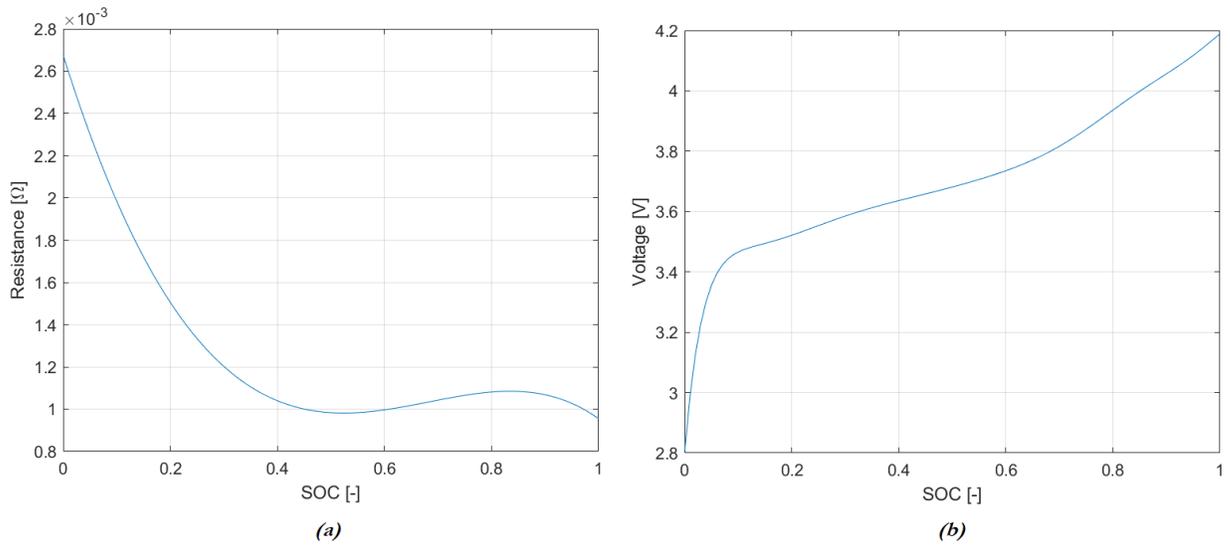


Figure 15 Single cell (a) Internal resistance in function of SOC, (b) Open circuit voltage in function of SOC

The rate of change of the state of charge is negative when the torque is needed to traction and it is positive when the torque is used for regenerative braking. This value is integrated and added to an initial state of charge value to obtain the instantaneous SOC at each time step.

Also, using the values of internal resistance and open circuit voltage it is possible to obtain the instantaneous load voltage for the selected battery, that has 99 cells in series, by the following equation:

$$V_L = V_{OC} - R_{int}I_L \quad (16)$$

where V_L and I_L are the load voltage and current, respectively.

To validate the correct behaviour of the obtained model, the simulation of a drive cycle is run. The selected cycle is WLTP class 3, applied to all the vehicles with a power-to-mass ratio higher than 34 W/kg , that has a length of 1800 s . The controller is a PI controller that sends acceleration and deceleration signal at each time step, comparing the speed of the drive cycle profile with the output speed of the model. The initial state of charge has been selected as 80% of battery full charge, that is the threshold that allows regenerative braking. The results in terms of speed profile is shown in Figure 16.

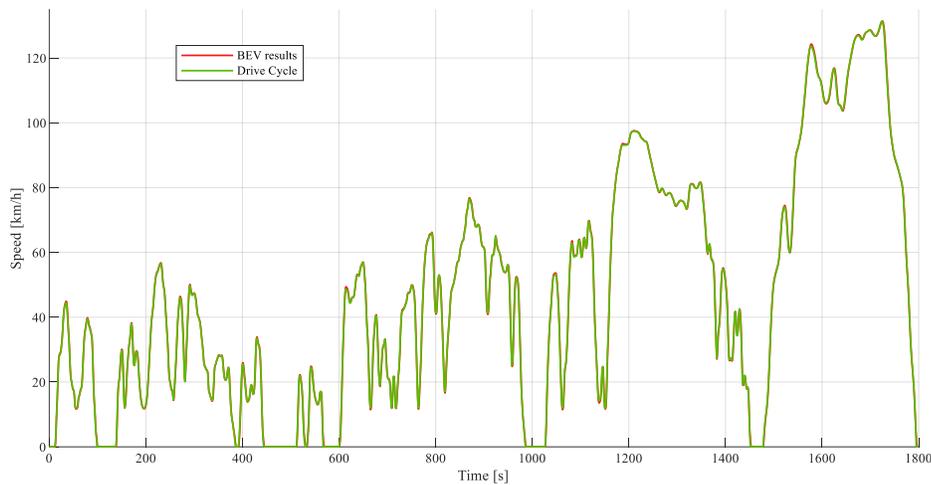


Figure 16 Speed profile of the drive cycle and simulated model

It is possible to see that the results of the simulated model are very close to the speed required from the drive cycle, with a small percentage error through the whole cycle. This result depends on the values of the proportional (K_p) and integral (K_I) constant that are selected in this case as a trade-off between the precision of results and simulation time.

A very important result is to obtain a faithful representation of the state of charge of the battery. The SOC is shown in the plot in Figure 17.

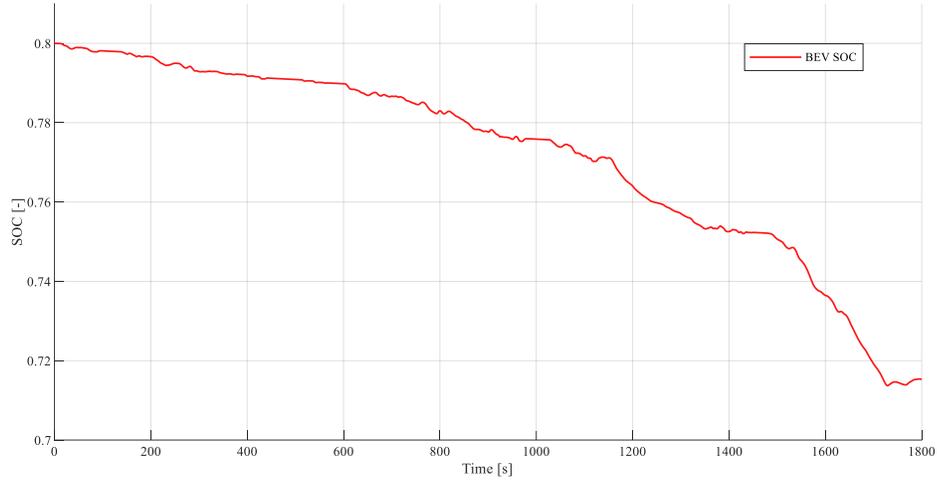


Figure 17 State of Charge of the vehicle's battery during the WLTP drive cycle

The trend of Figure 17 is satisfying, because it has high discharging when strong acceleration transient are needed. Also, during the decelerations, it is possible to observe an increase, even if small, of the state of charge thanks to regenerative braking.

The difference in magnitude of charging and discharging is possible to find using the power, torque and efficiency plots, shown in Figure 18, Figure 19, Figure 20 and Figure 21.

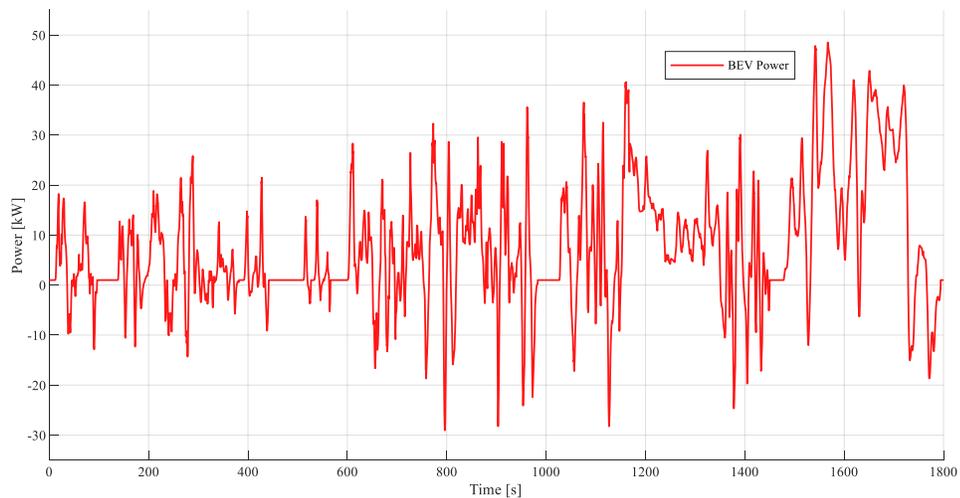


Figure 18 Power required/produced from the motor during the WLTP drive cycle

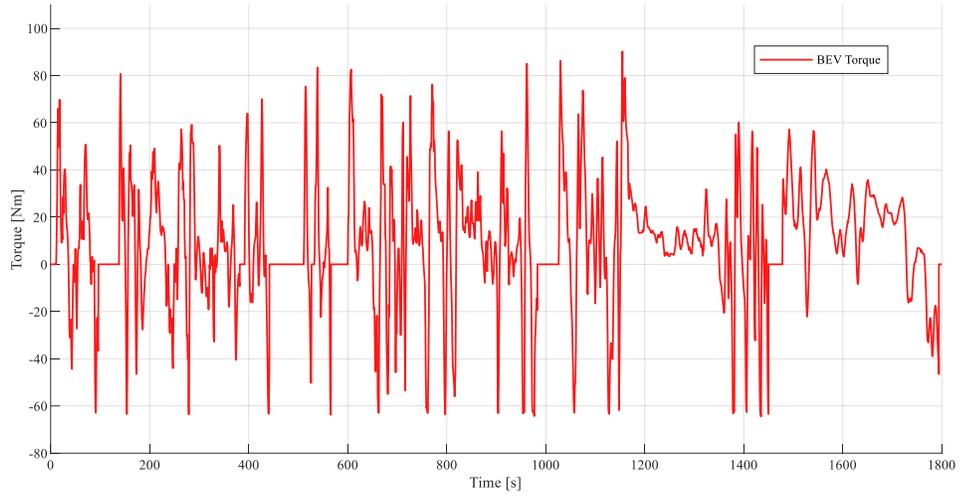


Figure 19 Torque of the motor during the WLTP drive cycle

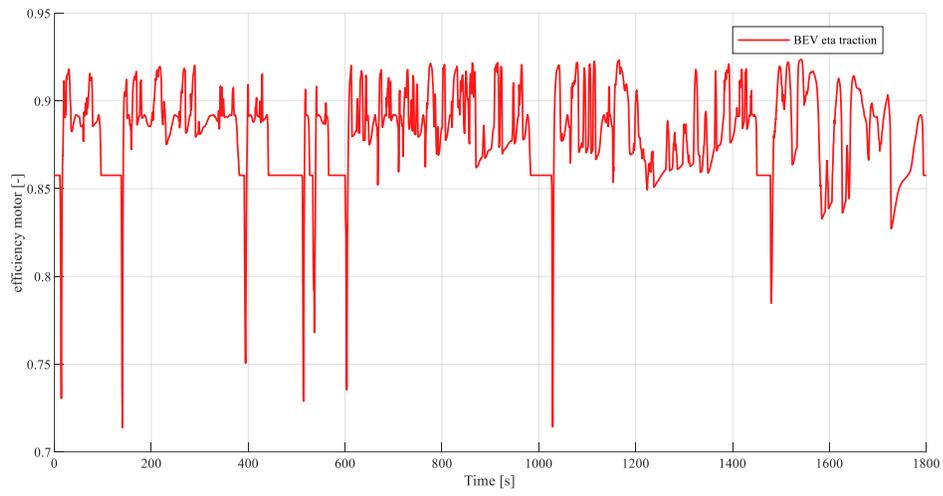


Figure 20 Efficiency of the electric motor in traction during the WLTP drive cycle

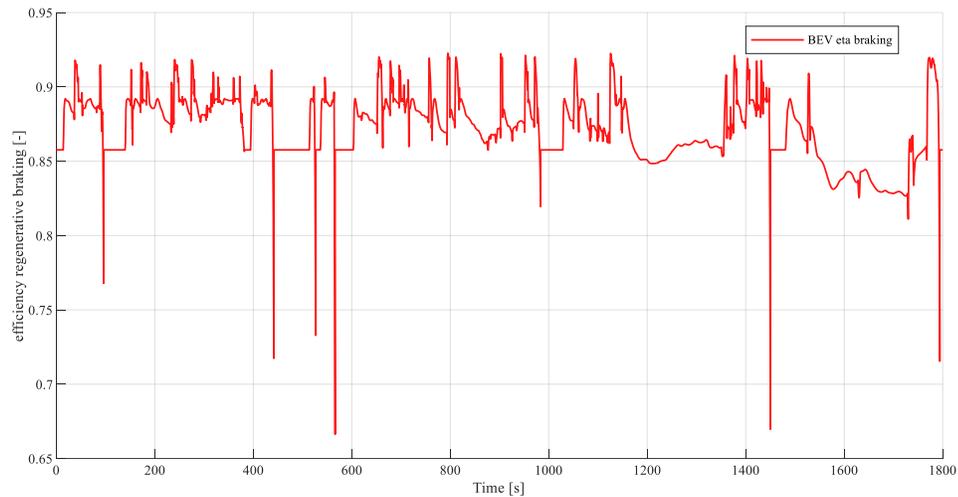


Figure 21 Efficiency of the electric motor while using regenerative braking during the WLTP drive cycle

In fact, looking at the torque plot it is possible to observe that the maximum torque required is nearly the same in absolute value both in traction and braking. However, the power required to the battery during the accelerations is greater than the power produced in braking thanks to regenerative braking (Figure 18). The reason of this trend is detectable in some different causes. First of all, the lower motor efficiency during braking compared to traction, even if the difference is not too large, as it is possible to see comparing the efficiencies in Figure 20 and Figure 21. Moreover, this difference is generated when, using a high torque during braking, a strong deceleration is produced which quickly brings to a small value of the angular speed and reduce the power generated by the electric motor.

This is a realistic behaviour because during braking only a small part of the energy can be recovered and send to the battery.

Looking at the torque (Figure 19) it is possible to see that maximum torque can always been supplied despite having only one gear. This is thanks the favourable characteristic of the electric motor that allow to provide the higher torque already at low speed. However, the high torque request in traction leads to a faster discharge of the battery.

All this is reflected also on the current flowing into the battery, as exhibited in Figure 22.

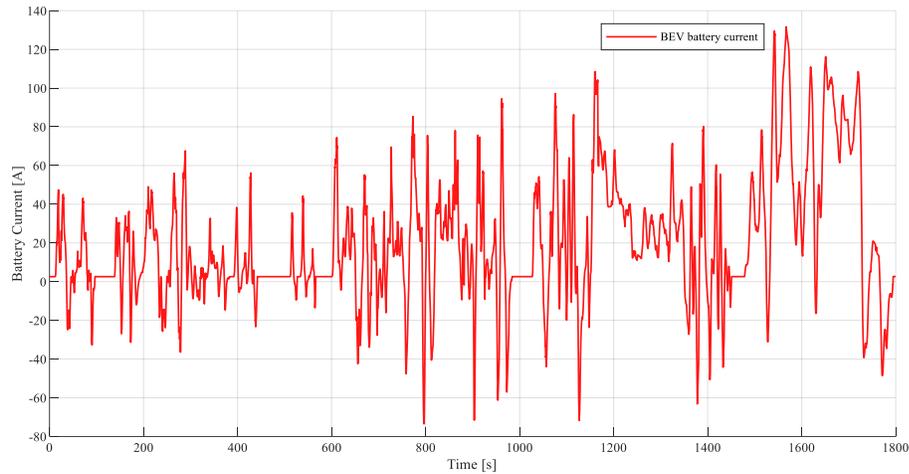


Figure 22 Battery current during the WLTP drive cycle

As expected the current flowing into the battery is higher during the extra-high phase of the WLTP drive cycle, that requires the maximum power to the motor and it is lower in the other phases and during braking. It is, also, influenced by the variable voltage of the battery that depends on the state of charge and decrease during discharging, as it is possible to see in Figure 23.

The voltage decreases and increases with the SOC and tends to decrease during traction and increase in braking. For example, it is possible to observe an increase in the output battery voltage with the strong deceleration at the end of the cycle, thanks to the energy recovered with regenerative braking that leads to a raise of the final state of charge.

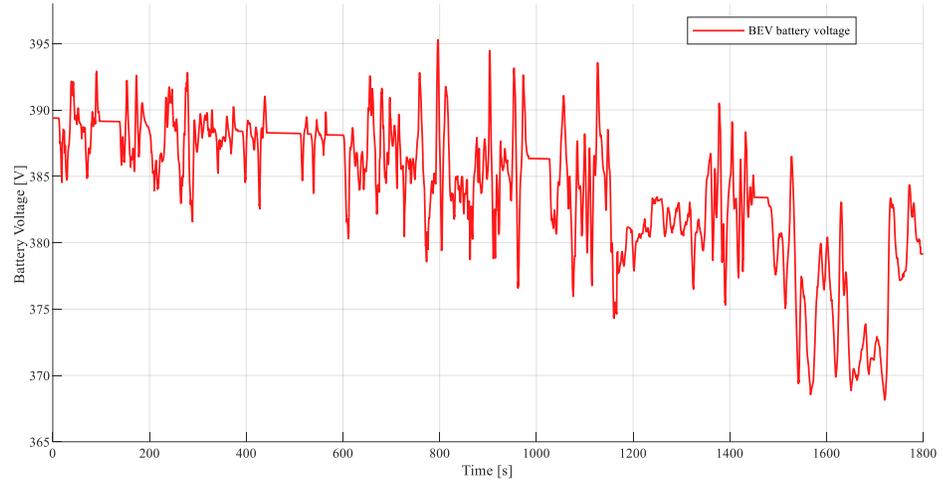


Figure 23 Battery voltage output during the WLTP drive cycle

5. MPC Toolbox

The model previously developed is transformed into a subsystem to implement the desired Model Predictive Control application. Using the MPC toolbox of MATLAB/Simulink, it is possible to build an MPC controller for several applications. As said before, the purpose of this case study is the developing of an electric vehicle equipped with Cooperative Adaptive Cruise Control (CACC) technology.

5.1 Cooperative Adaptive Cruise Control (CACC) [19]

Cooperative Adaptive Cruise Control is the combination of two concepts: automated speed control and cooperative element, such as V2V communication. It maintains close vehicle spacing, increasing highway lane capacity, but, in this circumstance, is not likely to maximize fuel efficiency under traffic conditions that require frequent speed changes. The main reasons that encourage CACC developing are improving traffic flow, safety and comfort. CACC is more attractive than conventional Adaptive Cruise Control (ACC) because the system behaviour can be more responsive to changes in the preceding vehicle speed and enables shorter following gaps. It represents a Level 1 automation on both SAE and NHTSA scales of autonomous driving, because it only provides longitudinal control of the vehicle, while the driver remains responsible of steering control and monitoring the driving environment.

Many different vehicle-follower speed control strategies have been proposed over the years, based on a wide variety of feedback control approaches. The most important are: Constant Distance Gap, Constant Time Gap and Constant-Safety-Factor Criterion. The strategy used in this work is the Constant Time Gap, the most closely to human driving, in which the distance between vehicles is proportional to their speed, plus a small fixed offset distance. The time gap criterion is described in terms of the time between when the rear bumper of the leading vehicle and the front bumper of the following one pass a fixed location on the roadway. The other two strategies consider, respectively, a constant distance, regardless the vehicle speed, and a inter-distance proportional to the square of the cruising speed. These two methods are not preferred because the Constant Distance Gap is too dangerous in case of emergency braking, requiring very high inter-vehicle distances, whereas the Constant-Safety-Factor Criterion is very severe trying to avoid collisions between more than two vehicles.

5.2 MPC Parameters

To properly develop a MPC controller several parameters are required:

- MPC structure (number of input, output, measured and unmeasured disturbances);
- Internal plant of the model;
- Sample time;
- Prediction horizon;
- Control horizon;
- Scale factors;
- Weights;
- Constraints;
- Robustness or aggressiveness of the controller.

Thanks to the MPC designer of the MPC toolbox it is possible to set all this factors.

5.2.1 MPC structure and internal plant model [20]

First of all, to define the structure of the MPC is required to select which and how many are the input and outputs of the system.

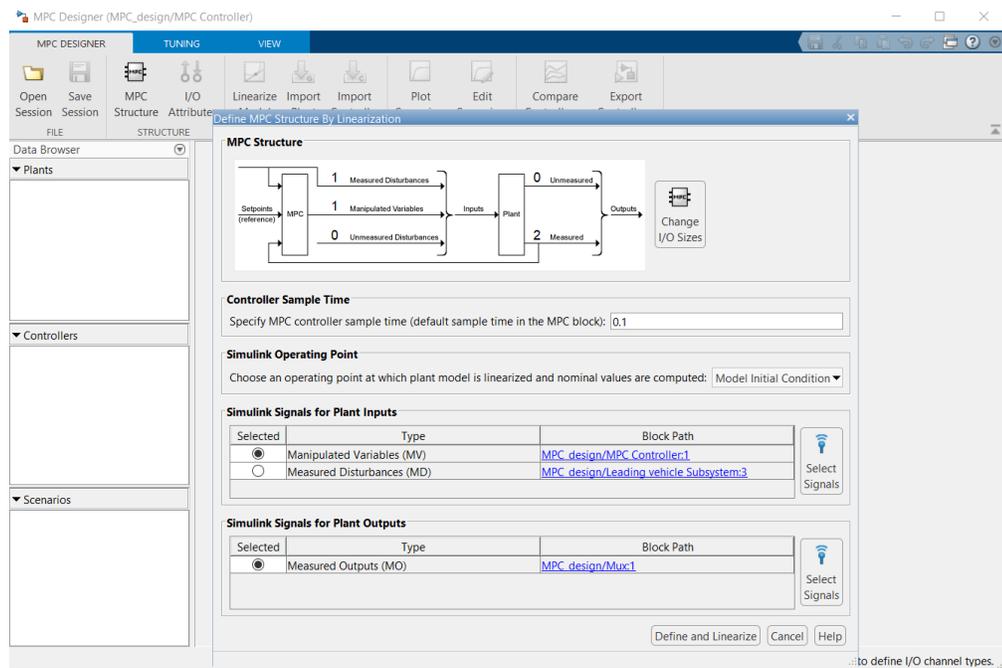


Figure 24 Define MPC structure and linearize the model

The measured outputs are the signals to be held at the desired setpoint, the manipulated variables are the signal MPC adjusts in order to achieve its control objectives and the measured disturbances are signals that the controller cannot adjust, but it uses them for feedforward compensation [21]. Then selecting the sample time and clicking the “Define and linearize” button in Figure 24, the MPC designer, using the Simulink Control Design, create a state space of the system internal to the controller. This state space model is linearized at a selected operating point (Model initial condition, snapshot time or steady-state operating point) and at which the nominal conditions are computed. In an ordinary MPC controller the plant model and the nominal conditions cannot vary during the simulation and that can cause prediction accuracy to degrade during the simulation if the characteristic of the system vary during time.

The state space plant model has the following formulation:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) + Du(k) \end{cases} \quad (17)$$

where:

- A, B, C, D are the state matrix;
- x is the states vector;
- u is the inputs vector;
- y is the outputs vector;
- k is the current time step.

This model is put in the MPC designer plant tab and is used to make predictions of possible future control actions and find optimal sequence.

5.2.2 Sample time, prediction horizon and control horizon [22]

Three primary parameters in MPC controllers, as said before, are sample time, prediction horizon and control horizon. On them rely the computational effort and the sharpness of the controller.

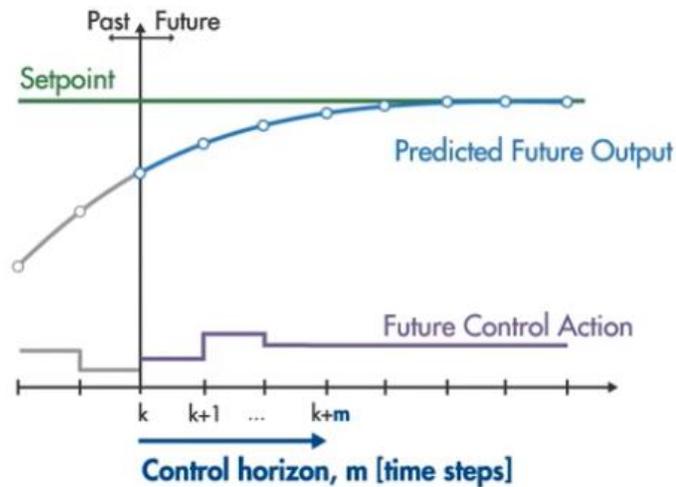


Figure 25 Sample time, prediction and control horizon

A recommended practice is to choose the sample time at the beginning of the controller design process, while tuning the other two parameters. Obviously, if the initial choice for the sample time appears poor it is possible to change it and then retune the other parameters. Qualitatively, as sample time decreases, rejection of unknown disturbance usually improves and then plateaus. The value at which performance plateaus depends on the plant dynamic characteristics. However, as sample time becomes small, the computational effort increases dramatically. Thus, the optimal choice is a balance of performance and computational effort.

The guidelines suggest to set sample time between 10% and 25% of your minimum desired closed-loop response time. For process control typically the sample time is greater than 1s, especially when MPC supervises lower-level single-loop controllers. Other applications, such as automotive or aerospace, can require sample time smaller than 1s.

Also for the prediction horizon is recommended to choose it early during control design and then hold it constant while tuning other control settings, such as weights. Rather, the value of the prediction horizon should be such that the controller is internally stable and anticipates constraint violations early enough to allow corrective action. Recommended practice is to increase prediction horizon until further increases have a minor impact on performance. Also, prediction horizons greater than 50 are rarely necessary unless the sample time is too small. To check if the controller is internally unstable it is possible to use the review design tab of the MPC

designer and click on the corresponding voice. If the prediction horizon is already large and the controller is unstable, the suggested actions are to increase the sample time, increase the cost function weights on the measured variable increments or modify the control horizon.

The control horizon is between 1 and the prediction horizon. Regardless of the control horizon chosen, when the controller operates, the optimized measured variable move at the beginning of the horizon is used and any others are discarded. The guidelines suggest to set the control horizon between 10 to 20% of the prediction horizon and having minimum 2 or 3 steps. In fact, is important to keep the control horizon smaller than the prediction horizon to promote faster computation and an internally stable controller (the second one is not guaranteed). Also, if the control horizon is too big some moves on the measured variables might not affect any of the plant outputs before the end of the prediction horizon.

5.2.3 Scale factors [23]

Recommended practice includes specification of scale factors for each plant input and output variable, which is especially important when certain variables have much larger or smaller magnitudes than others. The MPC controller divides each plant input and output signal by its scale factor to generate dimensionless signals. Choosing proper values for scale factors helps tuning the weights of input and output. Indeed, the correct choose of scale factors allow to focus on the relative priority of each term of the weights rather than a combination of priority and signal scale. It also improves numerical conditioning because when the values are scaled, round-off errors have less impact on calculations. To select the correct value for scale factors, the guidelines suggest three situations:

- If the signal has known bounds, use for scale factor the difference between the upper and lower limit;
- If the signal bounds are not known, consider running open-loop plant model simulations. It is possible to vary the inputs over their ranges, and record output signal spans;
- If neither of the above is possible, use the default value of 1.

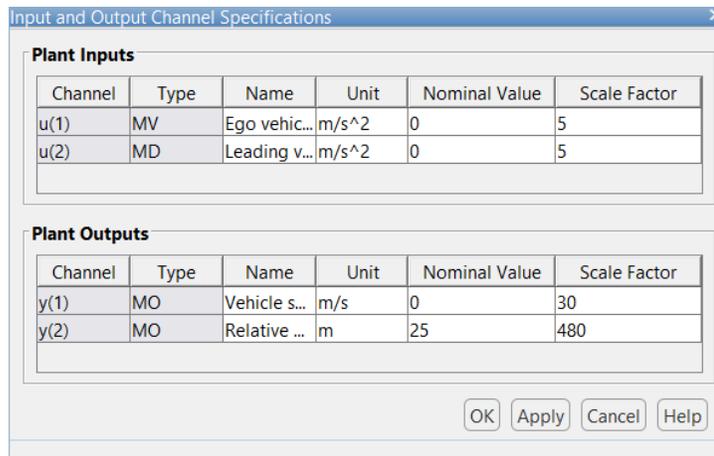


Figure 26 Scale factor dialog box

5.2.4 Constraints [24]

One of the most important benefits of the Model Predictive Control is to optimize the control sequence while considering constraints on the input and output. These are limits for the manipulated variables and the measured outputs that result from physical limits or limits imposed for the specific application by the controller designer.

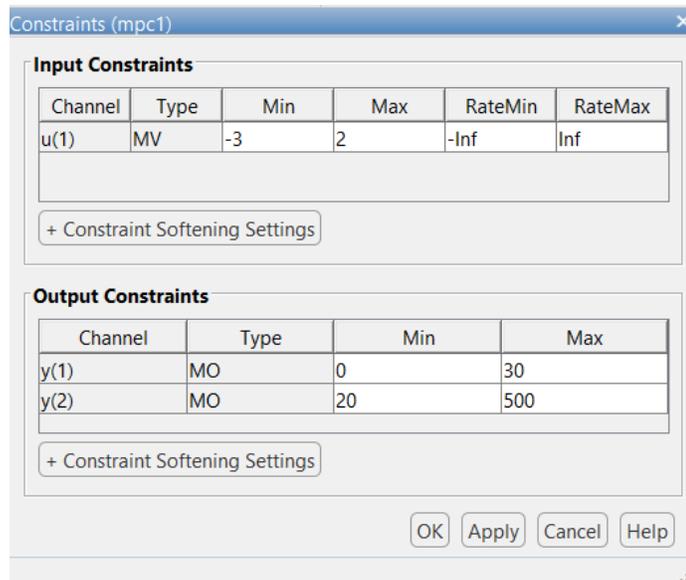


Figure 27 Constraint dialog box

They can be distinguished in hard or soft constraints: the hard constraints cannot be violated, while the soft constraints can be violated within certain tolerable interval. If it is mathematically

impossible to satisfy a hard constraint at a given control interval, k , the quadratic programming is infeasible. In this case, the controller returns an error status, and sets the manipulated variables to $u(k) = u(k-1)$, that is no change. If the condition leading to infeasibility is not resolved, infeasibility can continue indefinitely, leading to a loss of control. Typically the constraints on the manipulated variables are hard by default. Imposing hard constraints also on the outputs or on the rate of change of the manipulated variables may lead to infeasibility. So it is suggested to avoid this circumstance.

To soften a constraint is possible to set the corresponding equal concern for relaxation (ECR) value to a positive value (zero implies a hard constraint). The larger the ECR value, the more likely the controller will deem it optimal to violate the constraint in order to satisfy your other performance goals. To modify the ECR value it is needed to open the ECR box clicking on the “Constraint Softening Settings” tab shown in Figure 27. To indicate the relative magnitude of a tolerable violation, use the ECR parameter associated with each constraint. The guidelines recommend the following values for softening the constraints:

- 0 - No violation allowed (hard constraint)
- 0.05 - Very small violation allowed (nearly hard)
- 0.2 - Small violation allowed (quite hard)
- 1 - average softness
- 5 - greater-than-average violation allowed (quite soft)
- 20 - large violation allowed (very soft)

Sometimes disturbances and prediction errors can lead to unexpected constraint violations in a real system. Attempting to prevent these violations by making constraints harder often degrades controller performance.

5.2.5 Weights [25]

Model predictive control solves an optimization problem at each time interval. The solution determines the manipulated variables to be used in the system until the next instant. To achieve optimization, we want to minimize a cost function in which each input or output variable has its

own weight. The greater is the weight of the variable, the more the controller will tend to keep it close to its target in order to minimize the cost function.

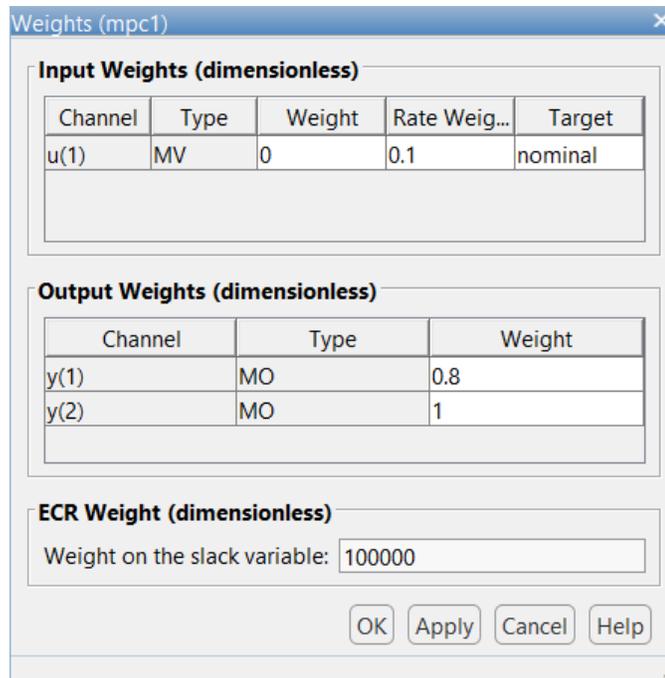


Figure 28 Weights dialog box

The weights of the measured outputs must be higher than zero, if you want to consider all those outputs in the cost function. On the contrary, the weights of the manipulated variables must be nonzero, only if these have targets. In this case it is possible to have a conflict between the manipulated variables and measured outputs tracking goals. So it is suggested to prioritize the tracking of measured outputs.

When choosing weights two important situations have to be considered:

- If the number of manipulated variables overcomes that of the measured outputs, there is an excess of degrees of freedom and the manipulated variables may drift even when the measured outputs are near their reference values. The preventive measures used to avoid this situation are to define targets for every existing manipulated variable or to increase the rate of change weight for these variables;
- If, on the contrary, the manipulated variables are less than the measured outputs, there are not enough degrees of freedom to keep all required measured outputs at a setpoint.

This can cause the incapacity of the controller to eliminate steady-state tracking error in all controlled outputs. It is possible to decrease the amount of error in a given output by making its weight value relatively large or to sacrifice an output by setting its weight value to zero, which should reduce errors in the other outputs. The guidelines for the weights priority are as follows:

- 0.05 - Low priority: Large tracking error acceptable;
- 0.2 - Below-average priority;
- 1 - Average priority (default);
- 5 - Above average priority;
- 20 - High priority: Small tracking error desired.

As well as the manipulated variables and the measured outputs, also the rate of change of the manipulated variables has an own weight. The simulation provides more robust controller performance, but poorer reference tracking the greater are the values of this weight.

At last it is important to consider the weight on the slack variable. This variable represents the violation of the constraints and the larger it is, with respect to input and output weights, the more the constraint violation is penalized. It has to be set such that the corresponding penalty is 1-2 orders of magnitude greater than the typical sum of the other three cost function terms. However, an excessively large slack variable weight distorts manipulated variables optimization, leading to inappropriate adjustments when constraint violations occur. To check for this, it is possible to display the cost function value during simulations. If its size increases by more than 2 orders of magnitude when a constraint violation occurs, consider decreasing this value.

5.2.6 Cost function [26]

The cost function is the scalar, nonnegative measure of controller that has to be minimized. The standard cost function is the sum of four terms as follows:

$$\begin{aligned}
J(z_k) = & \sum_{j=1}^{n_y} \sum_{i=1}^p \left\{ \frac{w_{i,j}^y}{s_j^y} [r_j(k+i|k) - y_j(k+i|k)] \right\}^2 \\
& + \sum_{j=1}^{n_u} \sum_{i=0}^{p-1} \left\{ \frac{w_{i,j}^u}{s_j^u} [u_j(k+i|k) - u_{j,target}(k+i|k)] \right\}^2 \\
& + \sum_{j=1}^{n_u} \sum_{i=0}^{p-1} \left\{ \frac{w_{i,j}^{\Delta u}}{s_j^u} [u_j(k+i|k) - u_j(k+i-1|k)] \right\}^2 + \rho_\varepsilon \varepsilon_k^2
\end{aligned} \tag{18}$$

where:

- z_k is the sequence selected from the controller;
- n_y and n_u are the number of measured outputs and manipulated variables, respectively;
- p is the prediction horizon;
- w are the weights of the variables or their rate of change;
- s are the scale factors of the variables;
- r is the reference for each output, while y is the output value;
- u e u_{target} are the manipulated variables and their possible target, respectively;
- ε_k e ρ_ε are the slack variable and its weight.

The first term is used to track the reference for the measured outputs. The controller receives reference values for the entire prediction horizon and using the state space model compute at the time step k the sequence of outputs, that depends only from the manipulated variables, the measured disturbance and state estimates. Then the value of this term is calculated. Only in some applications where the manipulated variable has a target, precomputed, the second term is considered. The third term is used to consider the increase or reduction of the rate of change of the manipulated variables. The last term contains the slack variable which quantifies the worst case constraint violation. The optimal solution will be the one that determines the sequence with the lowest value of the cost function.

An alternative expression of the cost function can be used by selecting it in the MPC block.

5.2.7 Robustness and aggressiveness

Once the weights have been chosen, it is possible to make the controller more aggressive or more robust.

Switching to more robust control reduces the weights of the outputs and measured variables and increases the weights of the rate of change of measured variables, which leads to relaxed control of outputs and more conservative control movements. The more aggressive control, on the other hand, leads to a better following performance of the references with greater variations of the manipulated variables.

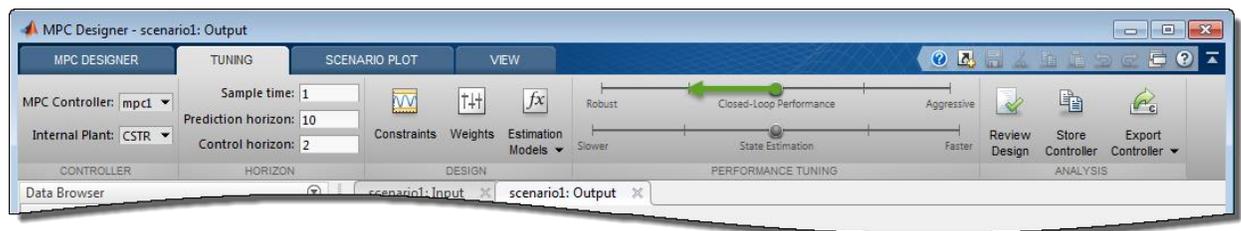


Figure 29 Closed-loop performance slider

5.3 Adaptive Cruise Control block [27]

Model Predictive Control Toolbox provides different blocks that simulate different operations of the vehicle, simplifying the design process. One of them is the Adaptive Cruise Control (ACC) block that it has been used to study how the parameters previously described (sample time, prediction and control horizon, robustness/aggressiveness, constraints, weights and scale factors) affect the controller.

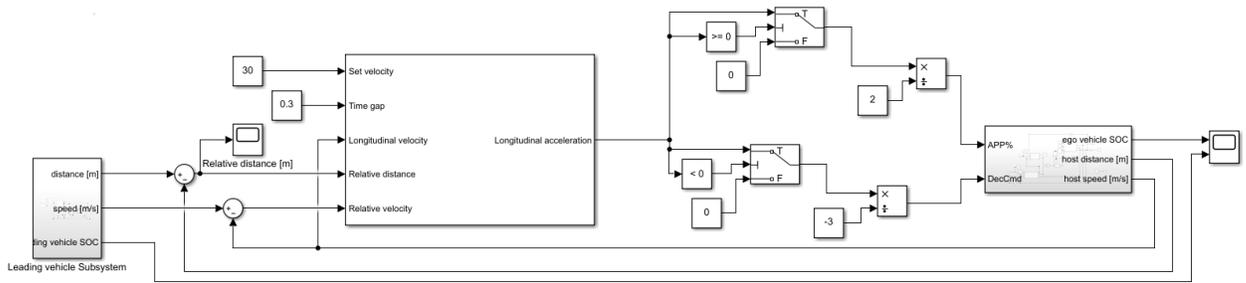


Figure 30 Adaptive Cruise Control structure in Simulink

The ACC block requires in input the set velocity, time gap, longitudinal velocity of the following vehicle, relative distance and speed between the two vehicles and gives the longitudinal acceleration as output. The set velocity is the speed required from the driver when there is no leading vehicle. The time gap is a parameter used to calculate the minimum safe following distance constraint, similarly to the CACC, as shown in the following expression:

$$D_{min} = D_{default} + T_{gap} \cdot V_{ego} \quad (19)$$

where:

- $D_{default}$ is the target relative distance between the ego and lead vehicles when the ego vehicle velocity is zero;
- T_{gap} is the time between when the rear bumper of the leading vehicle and the front bumper of the following vehicle pass a fixed location on the roadway;
- V_{ego} is the speed of the following vehicle.

With this criterion the distance between the two vehicles is proportional to their speed.

Relative speed and longitudinal speed are used as measured outputs and have as references the default spacing and the set speed, respectively, while the lead vehicle speed is computed through the relative speed and considered as a measured disturbance.

The plant model put in this block is represented by a simple transfer function which approximates the dynamics of the throttle body and vehicle inertia:

$$G = \frac{1}{s(0.5s + 1)} \quad (20)$$

Then this transfer function is transformed into the state space formulation and used by the optimizer to determine the best control sequence at each time step.

At last, the constraints on inputs and outputs are placed as follows:

- The maximum acceleration and deceleration are selected as 2 m/s^2 and -3 m/s^2 to not penalize the driving comfort during simulation. The comfort is a crucial objective therefore the constraints on acceleration are chosen as hard constraints;
- The maximum and minimum speed constraints are equal to 0 m/s and 40 m/s that is approximately the maximum speed that the vehicle can reach;
- The minimum distance between leading and following vehicle is calculated from the time gap expression shown above, while the maximum distance is unconstrained.

Then several simulations are run to tune the other parameters of the controller. In Figure 31, Figure 32 and Figure 33 are shown the results of the simulation that gives the best savings in terms of State of Charge (SOC) on WLTP drive cycle, obtained using the values shown in Table 3:

Sample time	0,5 s
Prediction horizon	150 steps
Robustness/Aggressiveness	0,8/1
Set velocity	30 m/s
Default spacing	25 m
Time gap	0,3 s
Initial SOC	0,8

Table 3 ACC simulation parameters

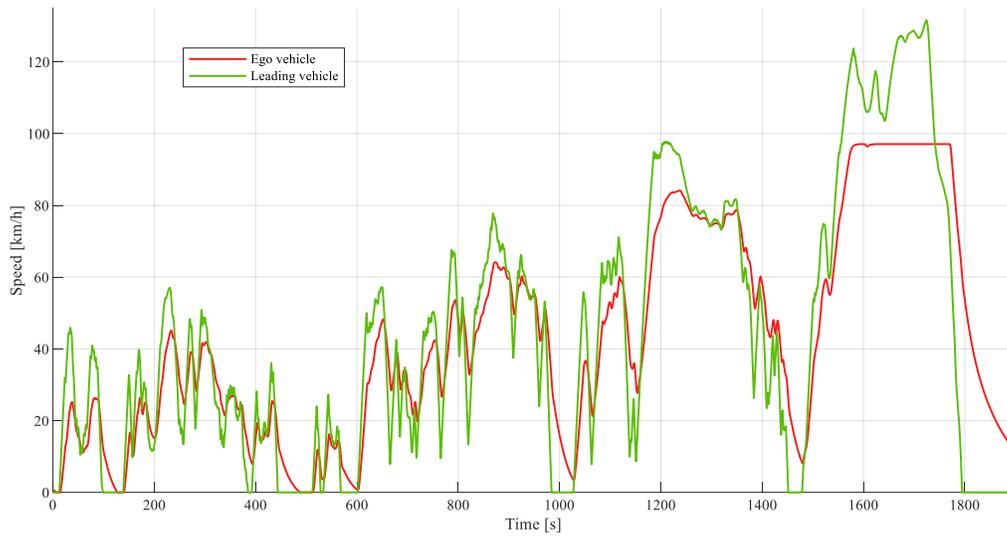


Figure 31 Speed profile of the leading and following vehicle with ACC on WLTP drive cycle

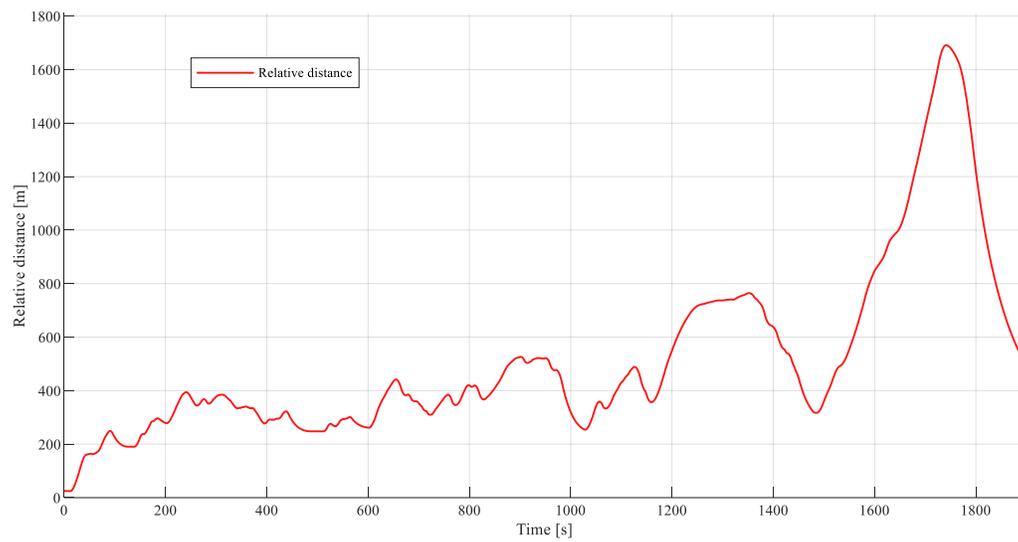


Figure 32 Relative distance between the leading and following vehicle with ACC on WLTP drive cycle

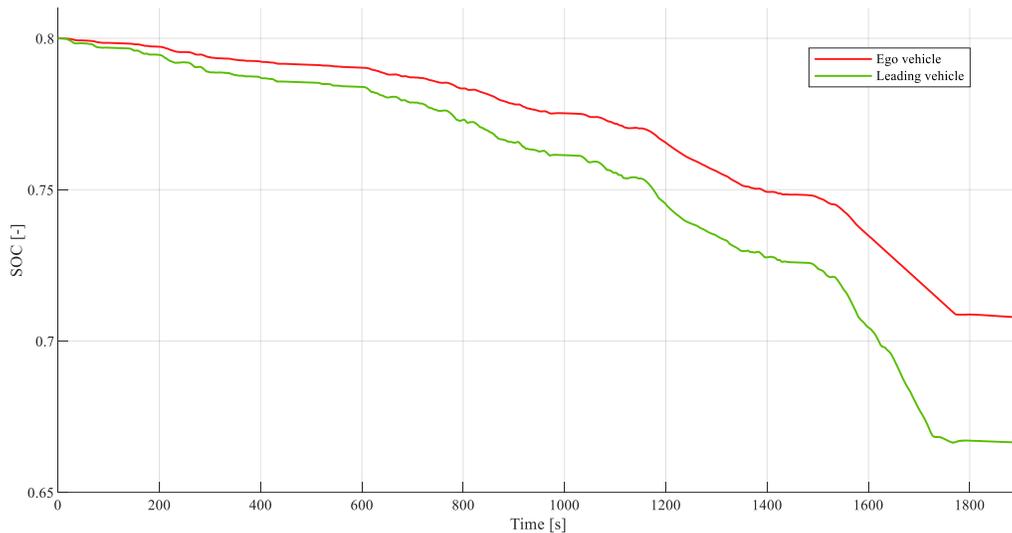


Figure 33 SOC profile of the leading and following vehicle with ACC on WLTP drive cycle

Watching at the results shown in figures, it is possible to see that the controller tries to smooth the drive cycle with less aggressive accelerations and decelerations. Indeed, the following vehicle has smoother acceleration ramps and never reaches the speed peaks of the leading vehicle. As well as during decelerations that don't follow perfectly those of the leading vehicle and often allow to reduce the gap between the two vehicles. All of this reflects on the SOC profile (Figure 33) giving excellent results and a final value for the SOC level of the following vehicle 6% higher than the leading one.

On the contrary, the relative distance increases quickly and very fast reaches values unacceptable, especially in the last part of the cycle when the following vehicle speed arrives at the constant value of the set speed required from the driver. In this case the relative distance is too large and any interaction is impossible between the two vehicles. Moreover, this behaviour will be totally unacceptable afterwards when the CACC is considered. Indeed, with this technology, the low value of the relative distance is a primary characteristic to allow V2V communication between the vehicles and the platooning.

5.4 Custom MPC

To achieve the goal of the work and design a controller that equip the vehicle with CACC, the blocks available from the MPC Toolbox are unsatisfactory. Thus, a custom MPC controller has been developed. The model used to design the controller is shown in Figure 34.

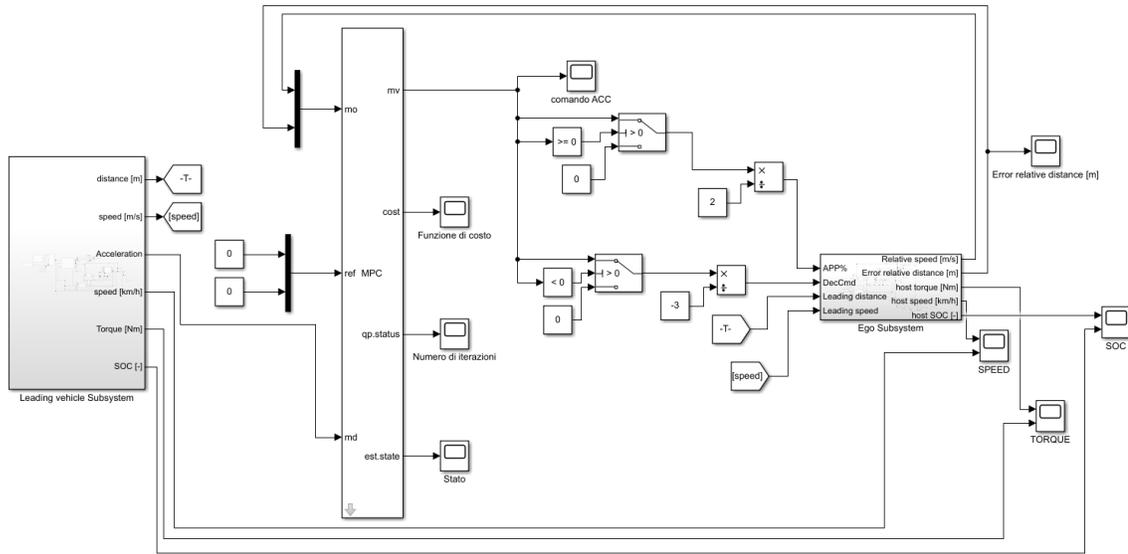


Figure 34 Custom MPC Simulink model

In this model the measured outputs are relative speed and spacing error, in order to maintain the relative distance near to a reference value computed as:

$$D_{ref} = D_{default} + T_{gap} * V_{follower} \quad (21)$$

where:

- $D_{default}$ is a constant minimum distance;
- T_{gap} is the time gap considered in the constant time gap criterion;
- $V_{follower}$ is the speed of the following vehicle.

Instead, the manipulated variable is the vehicle acceleration, while the acceleration of the preceding vehicle is considered as a measured disturbance, to enable feedforward compensation on this value. In this work, all position, speed and acceleration of the leading vehicle is provided to the following one through V2V communication.

After defining the inputs and outputs of the model, the sample time of 0,1 s has been chosen and using the Simulink control design, the model has been converted into state space form around model initial conditions. Precisely, the state space vector has been determined as:

$$x = \begin{bmatrix} T \\ s \\ \omega \\ v \end{bmatrix} \quad (22)$$

where:

- T is electric motor torque;
- s is vehicle position;
- ω is electric motor angular speed;
- v is speed of the vehicle.

At the same time, the state space matrices contains values obtained through approximation equations of the model.

Then, the prediction and control horizons have been selected. Respectively, the values of 50 and 15 steps have been considered adequate for the application.

At last, the constraints have been chosen both for input and outputs. As previously said, constraints are hard only on the inputs of the model while the outputs are soft. The selected constraints are:

- 2 m/s^2 and -3 m/s^2 as maximum and minimum acceleration, respectively. Like for ACC, this choice has been driven by the fulfilment of a good drive comfort for the passengers;
- 10 m/s and -10 m/s for maximum and minimum relative speed;
- 5 m and -5 for maximum and minimum spacing error, asking the controller to keep a small error between real and reference relative speed;
- Default value of 100000, if an input or output exceeds selected constraints.

Defined all the parameters required by the controller has been run several simulations, whose results are shown in Figure 35 and Figure 36.

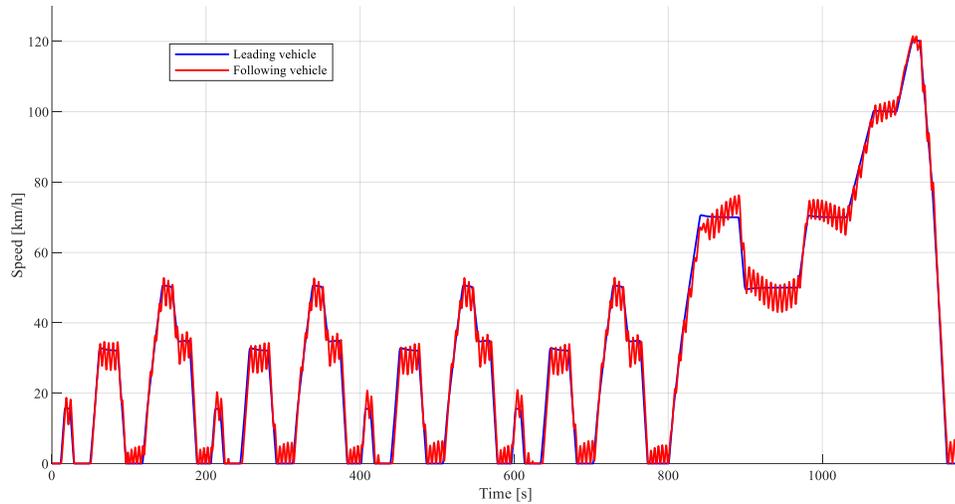


Figure 35 Speed profile of leading and following vehicle using custom MPC

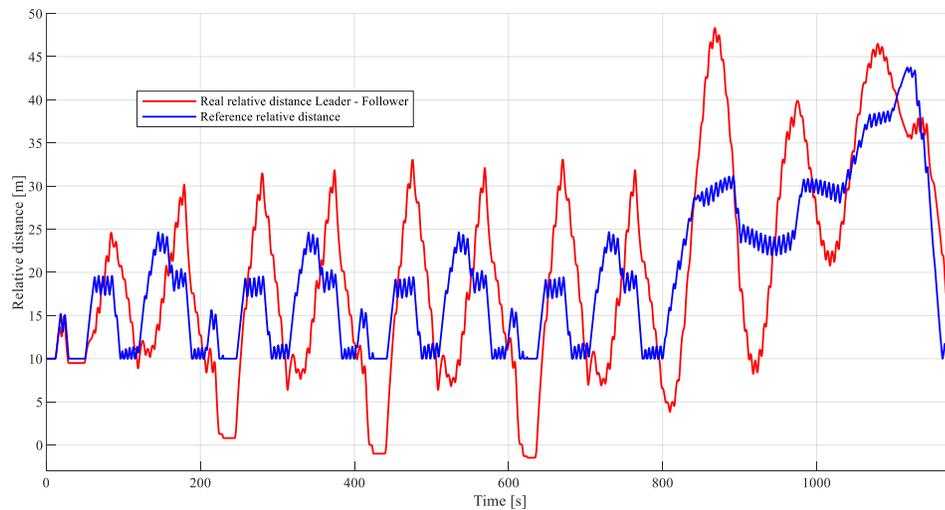


Figure 36 Real and reference relative distance using custom MPC

These results exhibited are unfeasible for the strong fluctuations shown in both the plots. In fact, the requests of low spacing error, comfort, platooning and safety demanded by CACC are not achieved.

The speed profile of the following vehicle fluctuate close to reference value for the whole cycle, especially when the speed of the leader is constant. This profile originates from an acceleration profile with strong oscillations, that correspond to a similar torque profile and power profile. All of this, cause a strong decrease of the state of charge (SOC) of the battery, that is totally unrealistic.

Referring to MPC Review this issue may be caused by an excess of measured outputs and, therefore, may not be enough degrees of freedom. Another cause of the problem may be that the plant model used by the controller is linearized around model initial conditions and , to produce proper results, probably, the model has to be recomputed at each time step using an Adaptive Model Predictive Control.

Otherwise, looking at the relative distance, not only the reference distance profile is incorrect due to speed fluctuations, also the real relative distance is very distant from its reference and during some time steps is even negative. Negative relative distance is a totally unrealistic value because means that there has been a collision between the vehicles and the controller have not worked properly.

Hence, to achieve the goal of this study, some changes in the controller are required. As previously said, the issue may be caused by the lack of degrees of freedom or because the model is defined near the initial conditions. So, an Adaptive Model Predictive Control is designed in order to resolve the issue.

6. Adaptive Model Predictive Control [28]

MPC control predicts future behaviour using a linear-time-invariant (LTI) dynamic model. In practice, such predictions are never exact, and a key tuning objective is to make MPC insensitive to prediction errors. In many applications, this approach is sufficient for robust controller performance. If the plant is strongly nonlinear or its characteristics vary dramatically with time, LTI prediction accuracy might degrade so much that MPC performance becomes unacceptable. Adaptive Model Predictive Control can address this degradation by adapting the prediction model for changing operating conditions. Indeed, Adaptive MPC at each time step recomputes the state matrices and the nominal conditions in the new operating point.

To implement Adaptive MPC in MATLAB/Simulink, MPC Toolbox provides a block that is similar to the conventional MPC block, but includes an extra input port called “model” where the updated state space model signal is sent. An example of Adaptive MPC is shown in Figure 37.

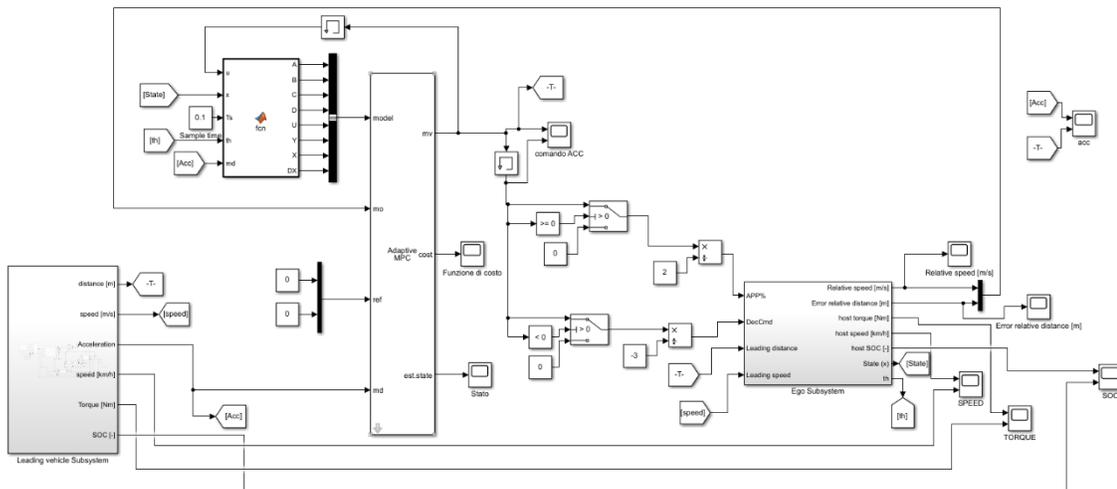


Figure 37 Adaptive Model Predictive Control design

Into the Adaptive MPC block the mpc previously developed using MPC design dialog box is loaded, after exporting it from the custom MPC. Indeed, it is possible to load the mpc into the workspace or define it in the command window and then, through the Adaptive MPC block use it. Figure 38 shows how the Adaptive MPC block recalls the previously defined mpc.

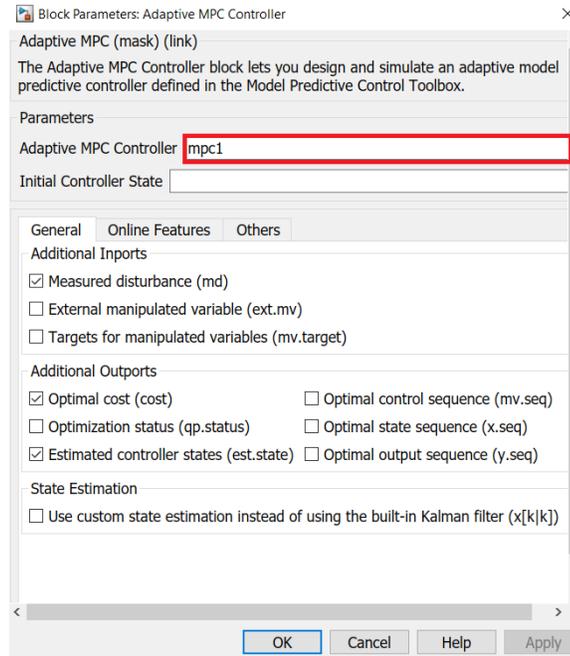


Figure 38 Adaptive MPC block mask

Then, this model is updated at each time step using a Function block of Simulink, that calls back a MATLAB function. Through this function the state matrices are recomputed, considering the longitudinal motion relationship between the leader and the following vehicle. The spacing error, one of the two measured outputs, is defined, considering *ith* following vehicle, as [29]:

$$\delta_i = \Delta s_{real,i} - \Delta s_{ref,i} \quad (23)$$

where:

- δ_i is the spacing error;
- $\Delta s_{real,i}$ is the real distance between *ith* vehicle and the preceding one, expressed as $\Delta s_{real,i} = s_{i-1} - s_i$, where s_i is the position of the *ith* vehicle and s_{i-1} is the position of the preceding one;
- $\Delta s_{ref,i}$ is the desired relative distance among the vehicles, as defined in the constant time gap policy, and computed as $\Delta s_{ref,i} = d_0 + t_h v_i$, where d_0 is the default standstill spacing, t_h the time gap and v_i the speed of the *ith* following vehicle.

Thus, the linearized dynamics of the *ith* following vehicle are modelled as:

$$\begin{cases} \dot{\delta} = v_{i-1} - v_i - t_h \dot{v}_i \\ \dot{v} = a_i \\ \Delta \dot{v}_i = a_{i-1} - a_i \\ \dot{a}_i = -\frac{a_i}{\tau_i} + \frac{u_i}{\tau_i} \end{cases} \quad (24)$$

where:

- v_{i-1} is the speed of the preceding vehicle;
- a_i and a_{i-1} are the acceleration of the i th vehicle and the preceding one, respectively;
- Δv_i is the relative speed between leading and following vehicle;
- τ_i is a time constant of the engine;
- u_i is the manipulated variable produced by the MPC controller.

To enable the computation of the input signal u_i , the system continuous time dynamics must be discretized using the Euler forward method with a time step of T_s :

$$\begin{cases} \delta_i(k+1) = \delta_i(k) + \Delta v_i(k)T_s - a_i(k)t_h T_s \\ v_i(k+1) = v_i(k) + a_i(k)T_s \\ \Delta v_i(k+1) = \Delta v_i(k) - a_i(k)T_s + a_{i-1}(k)T_s \\ a_i(k+1) = \left(1 - \frac{T_s}{\tau_i}\right) a_i(k) + \frac{u_i(k)T_s}{\tau_i} \end{cases} \quad (25)$$

where k is the current time step.

Hence, rewriting these discretized equations in a state space formulation, the state matrices and state and input vectors turn into:

$$A_i = \begin{bmatrix} 1 & 0 & T_s & -t_h \cdot T_s \\ 0 & 1 & 0 & T_s \\ 0 & 0 & 1 & -T_s \\ 0 & 0 & 0 & \left(1 - \frac{T_s}{\tau_i}\right) \end{bmatrix}; B_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & T_s \\ \frac{T_s}{\tau_i} & 0 \end{bmatrix}; C_i = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}; D_i = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (26)$$

$$x_i = \begin{bmatrix} \delta_i \\ v_i \\ \Delta v_i \\ a_i \end{bmatrix} \quad u_i = \begin{bmatrix} u_i \\ a_{i-1} \end{bmatrix} \quad (27)$$

These matrices are the output of the Function block, as shown in Figure 39.

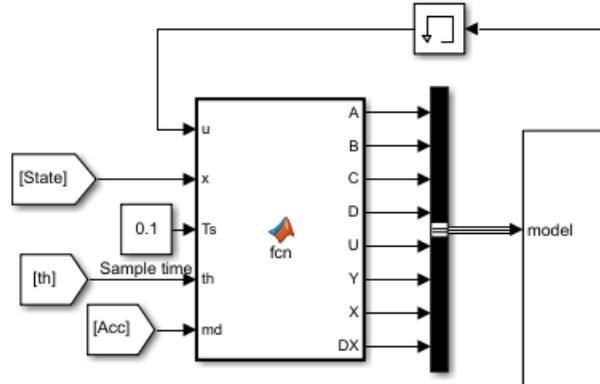


Figure 39 Simulink function block

Indeed, the Function block to work properly takes in input the state vector, the input vector, sample time of the MPC controller and time gap, while gives as outputs the state matrices and the updated nominal points. All the outputs have to be collected into a bus and sent to the Adaptive MPC, as required from MATLAB guidelines [30]. Considering this state space vehicle model, the controller finds the optimal input sequence $u_{opt}(k)$ minimizing the cost function, showed below:

$$\begin{aligned}
 J(k) = & \sum_{i=1}^p \{w_{\delta}[\delta(k+i|k) - 0]\}^2 + \sum_{i=1}^p \{w_{\Delta v}[\Delta v(k+i|k) - 0]\}^2 \\
 & + \sum_{i=0}^{p-1} \{w_{\Delta u}[u(k+i|k) - u(k+i-1|k)]\}^2 + \rho_{\varepsilon} \varepsilon_k^2
 \end{aligned} \tag{28}$$

where:

- p is the prediction horizon;
- w_{δ} , $w_{\Delta v}$ and $w_{\Delta u}$ are the weights assigned to the terms of the function considering spacing error, relative speed and acceleration increment, respectively;
- ε_k and ρ_{ε} are the slack variable, used to quantify the worst-case constraint violation, and its weight, respectively.

Therefore, the optimal control problem formulation is as follows:

$$\begin{aligned}
& \mathbf{u}_{opt}(k) = \arg \min_{\mathbf{u}} J(k) \\
& \text{subject to } \begin{cases} \delta_{min} \leq \delta(k+i|k) \leq \delta_{max} \\ \Delta v_{min} \leq \Delta v(k+i|k) \leq \Delta v_{max} \\ u_{min} \leq u(k+i|k) \leq u_{max} \end{cases} \quad i = 1, \dots, p
\end{aligned} \tag{29}$$

where δ_{min} , δ_{max} , Δv_{min} , Δv_{max} , u_{min} and u_{max} are the constraints on spacing error, relative speed and acceleration, respectively.

Then, the first element of the obtained control sequence $\mathbf{u}_{opt}(k)$ is applied to the system and the operation is repeated for the next time step shifting the receding horizon.

6.1 Validation of the controller on different driving cycles

To validate the so obtained controller, several simulations on different drive cycles are run. The results are considered adequate, if the following vehicle has a spacing error smaller than the set constraints, generating the platoon of vehicles, there are enhancements on comfort and the selected drive cycle is roughly duplicated by the flowing vehicle.

The parameters selected for the Adaptive MPC are the same for all the drive cycles selected. These parameters, collected in , are a trade-off between low computational load and high fidelity results.

Parameters	Values	Parameters	Values
T_s	0,1 s	Δv_{min}	-10 m/s
p	100 steps	Δv_{max}	10 m/s
m	25 steps	u_{min}	-3 m/s ²
$d_{default}$	10 m	u_{max}	2 m/s ²
t_h	1 s	w_δ	1
τ	0,1 s	$w_{\Delta v}$	1
δ_{min}	-5 m	$w_{\Delta u}$	0,1
δ_{max}	5 m	w_{ECR}	100000

Table 4 Adaptive MPC parameters

where:

- T_s is the sample time, while the fixed-step size is 0.01 s;
- p and m are, respectively, the prediction and the control horizon;

- $d_{default}$ is the initial inter-vehicular distance between two vehicles;
- t_h is the time gap between the leader and follower.

The selected drive cycles are FTP72, WLTP and US06 that are three drive cycles representative of realistic driving conditions and with strong acceleration and deceleration ramps. Furthermore, considering the new pollutants legislative framework, a Random drive cycle of 2751 seconds is tested. This cycle has been generated by the software Random Cycle Generator developed by TNO [31]. In the next subparagraphs the plot of speed, SOC, relative distance and spacing error are shown for these drive cycles.

6.1.1 FTP72

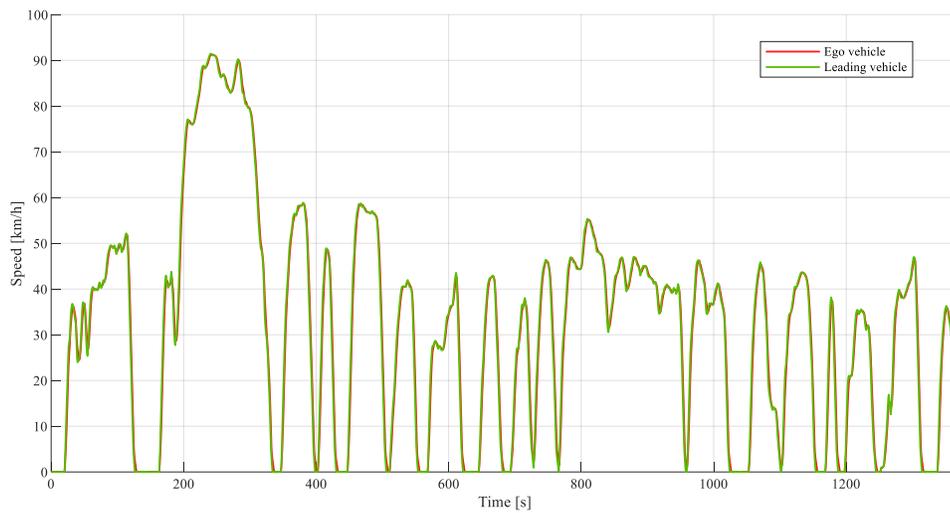


Figure 40 Leading and Ego vehicle speed on FTP72

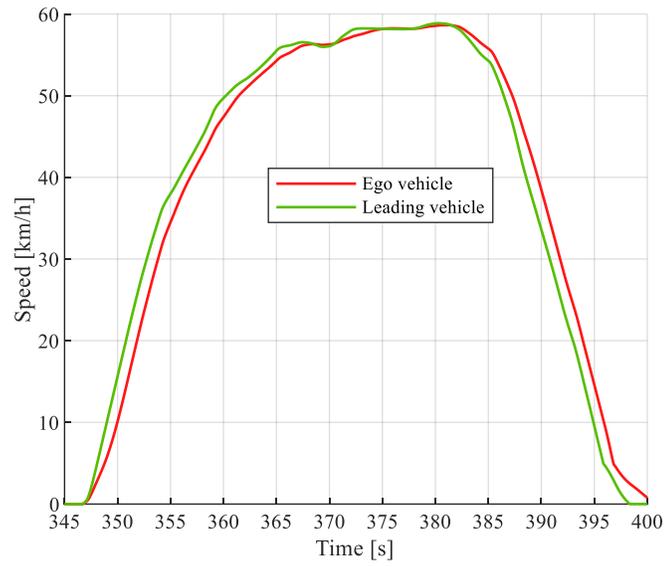


Figure 41 Zoom on vehicles speed between second 345 to 400 on FTP72

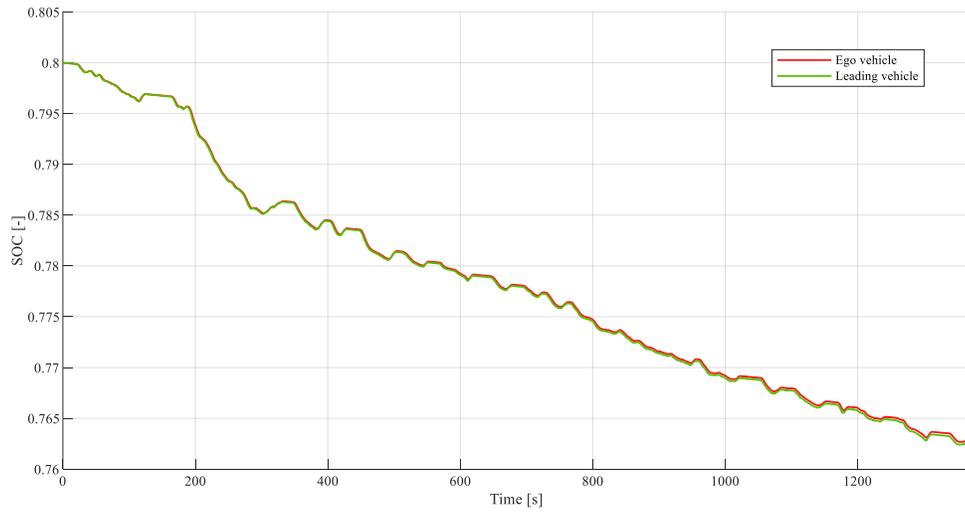


Figure 42 Leading and Ego vehicle SOC on FTP72

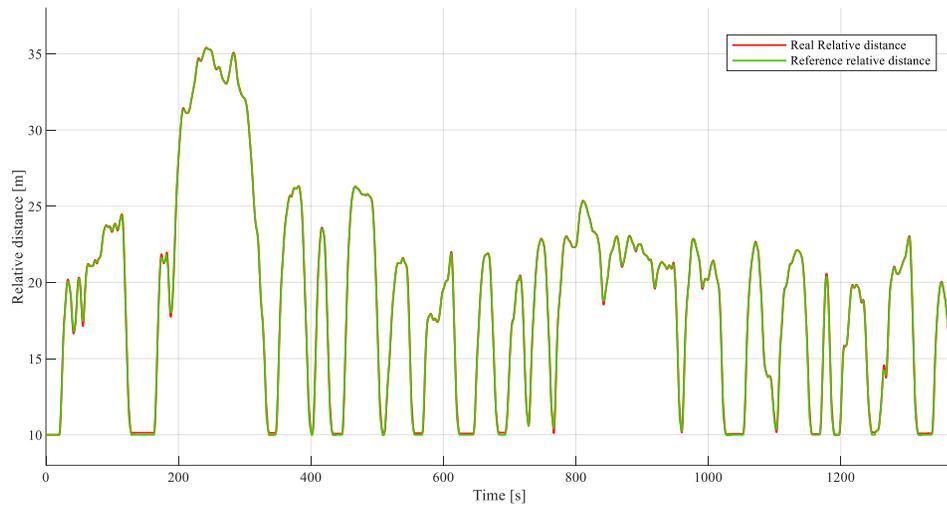


Figure 43 Required and real relative distance on FTP72

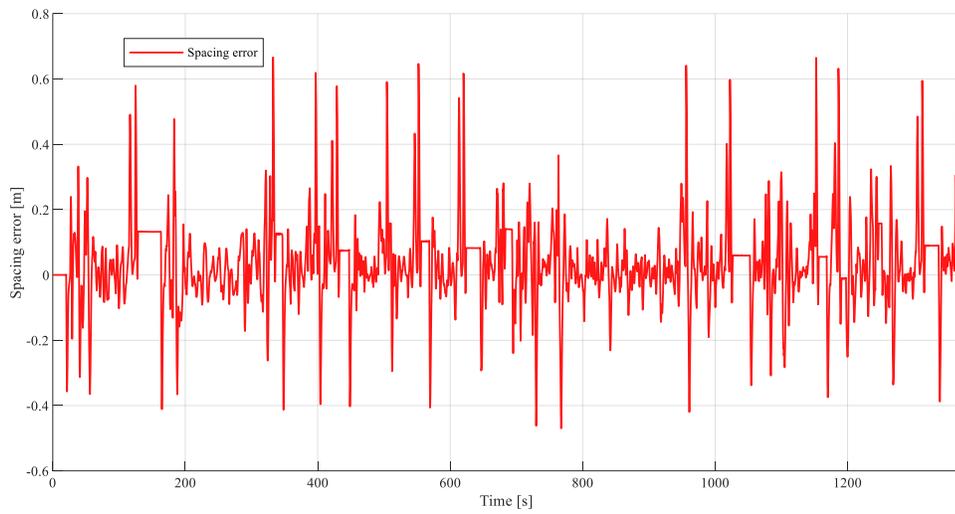


Figure 44 Spacing error on FTP72

6.1.2 WLTP

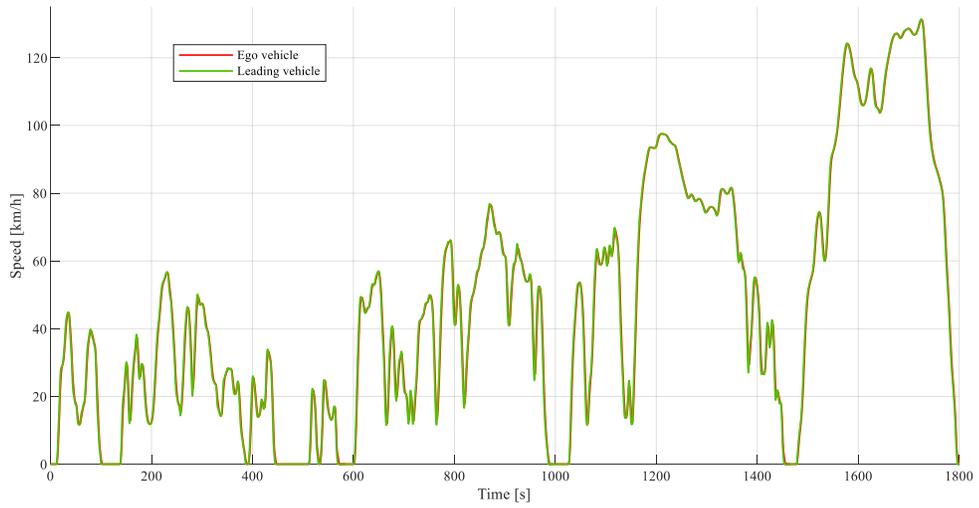


Figure 45 Leading and Ego vehicle speed on WLTP

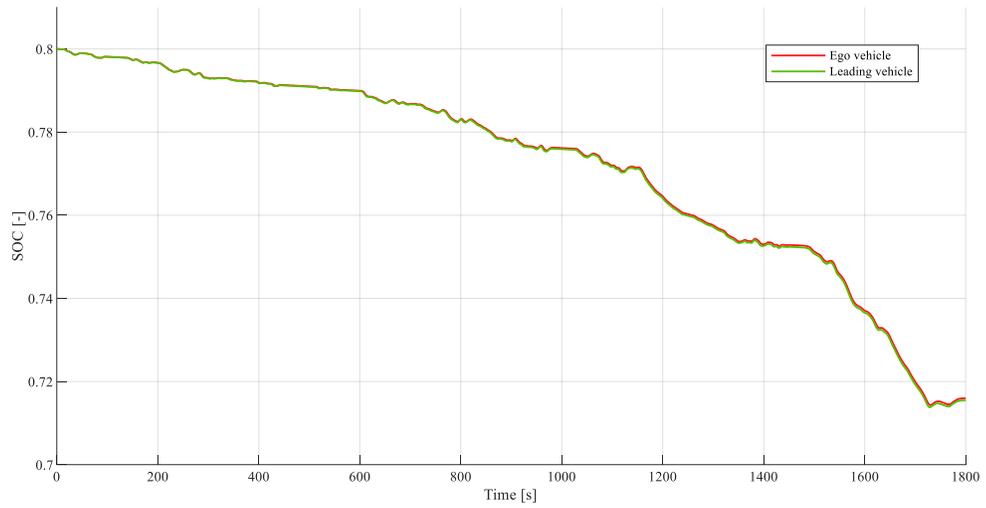


Figure 46 Leading and Ego vehicle SOC on WLTP

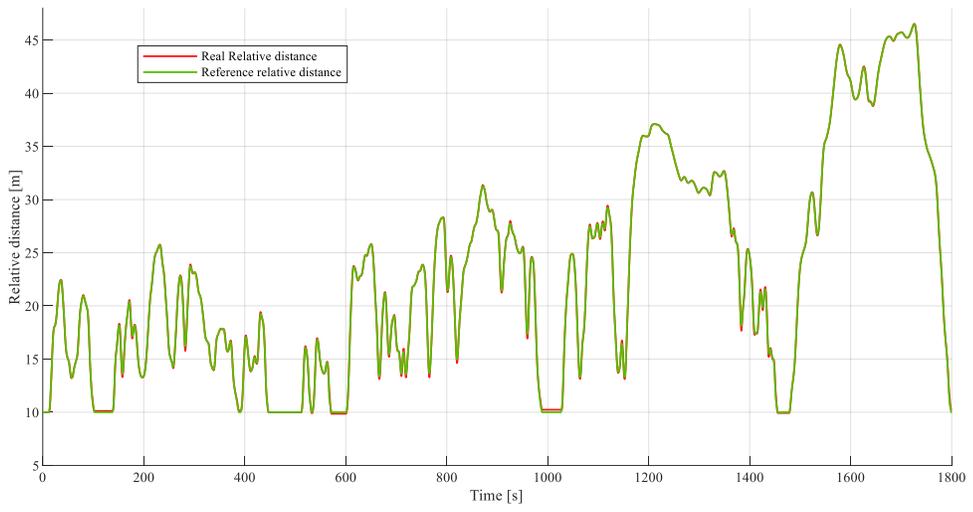


Figure 47 Reference and real relative distance on WLTP

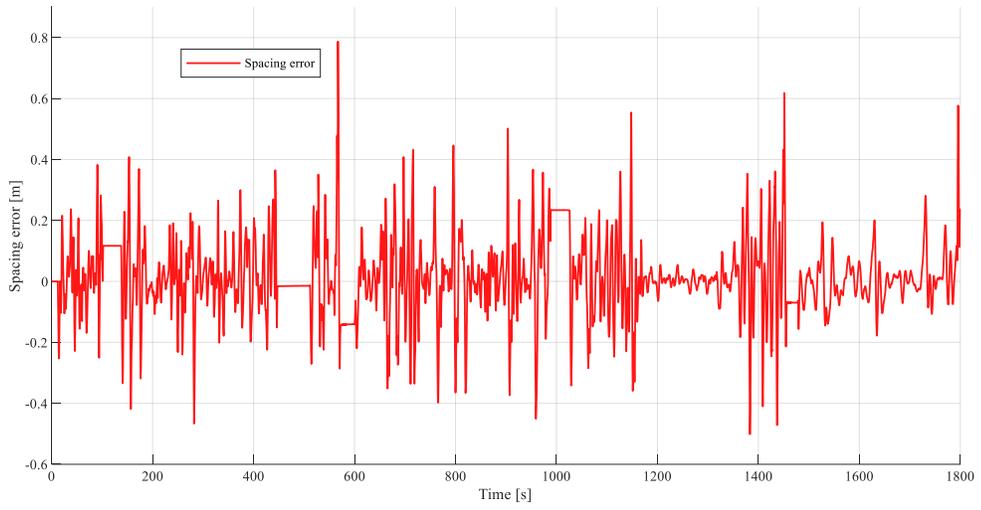


Figure 48 Spacing error on WLTP

6.1.3 US06

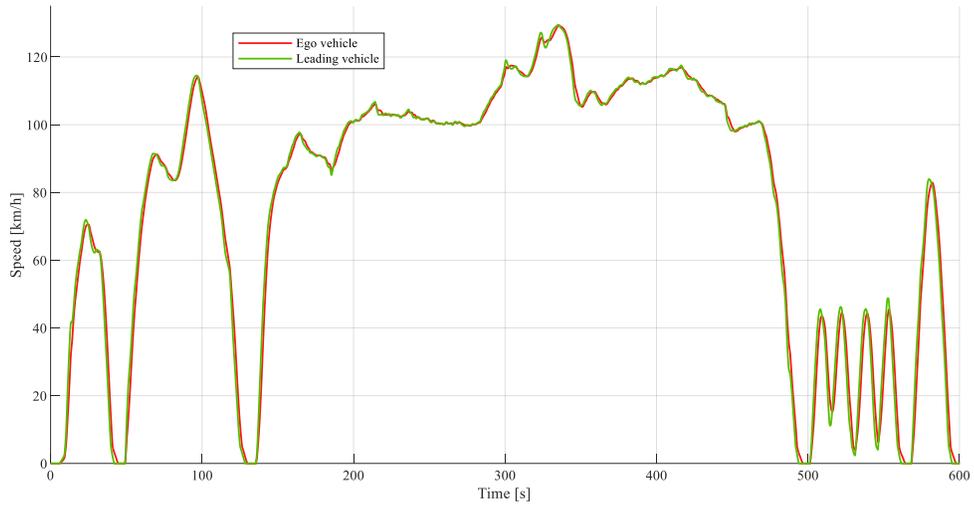


Figure 49 Leading and Ego vehicle speed on US06

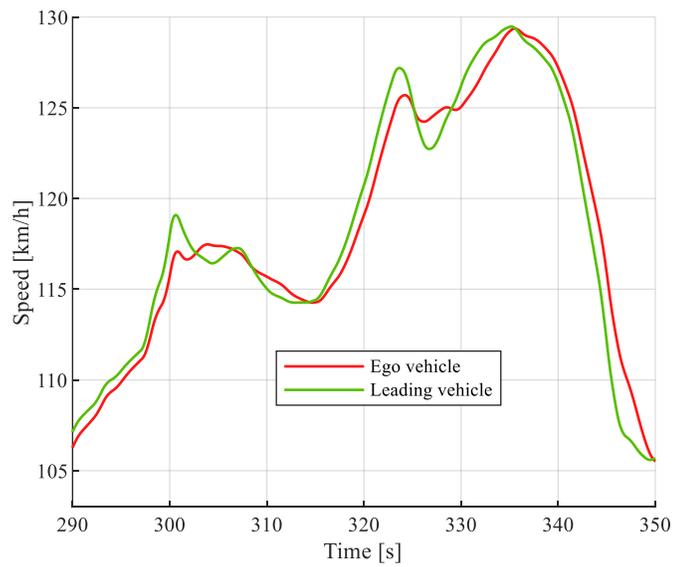


Figure 50 Zoom on vehicles speed between second 290 to 350 on US06

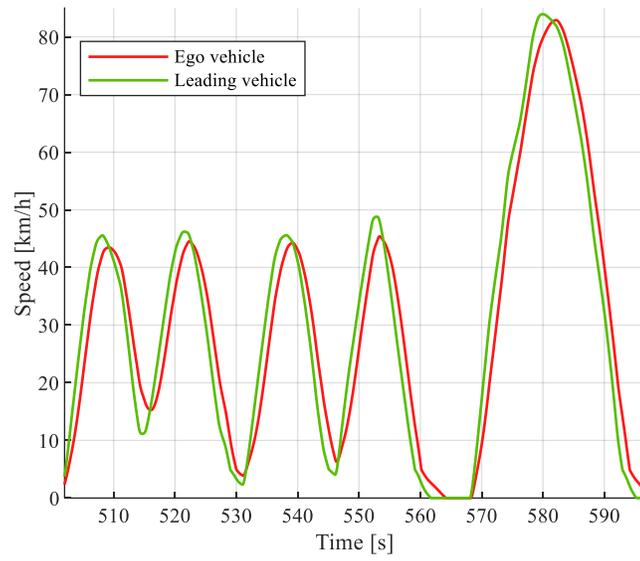


Figure 51 Zoom on vehicles speed between second 503 to 597 on US06

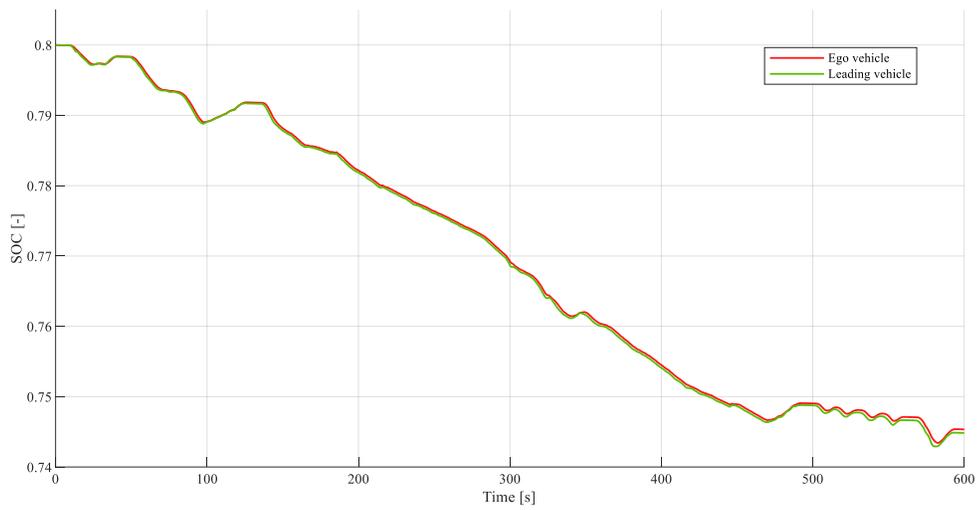


Figure 52 Leading and Ego vehicle SOC on US06

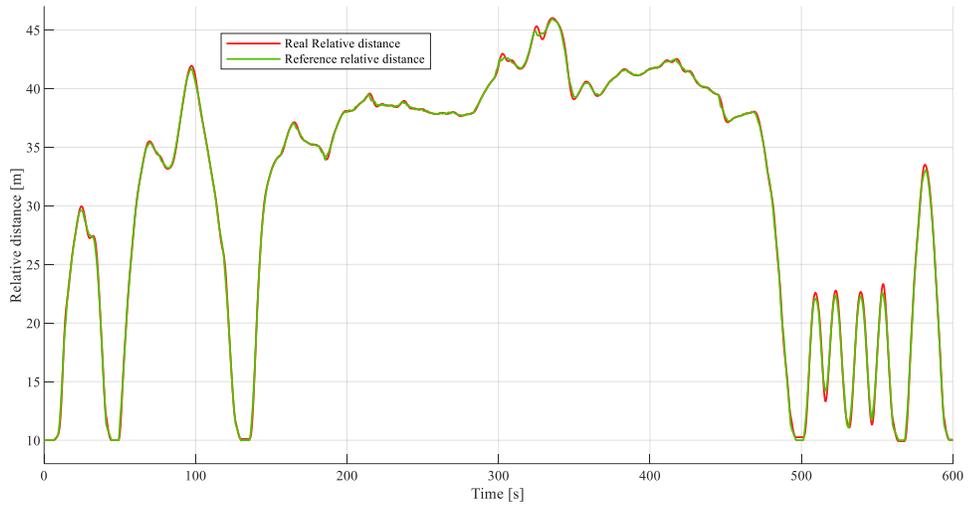


Figure 53 Real and reference relative distance on US06

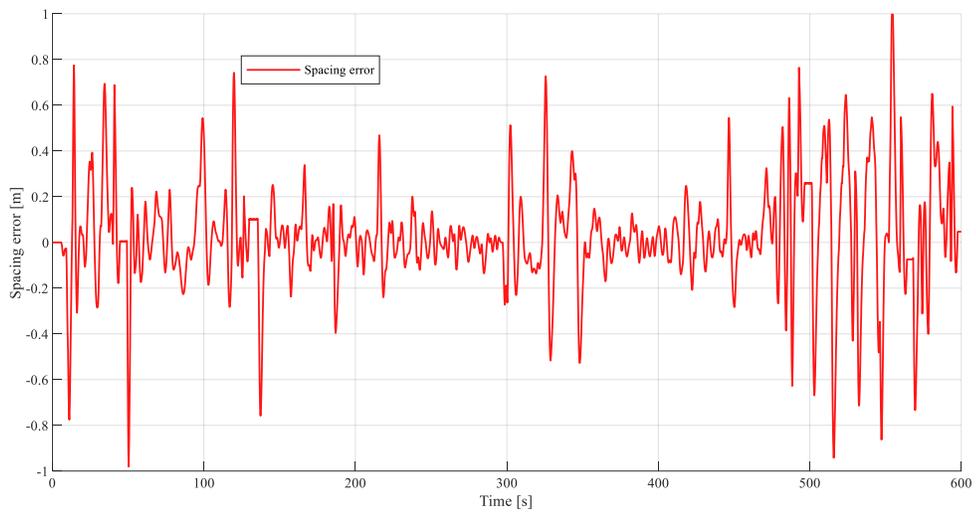


Figure 54 Spacing error on US06

6.1.4 Random cycle

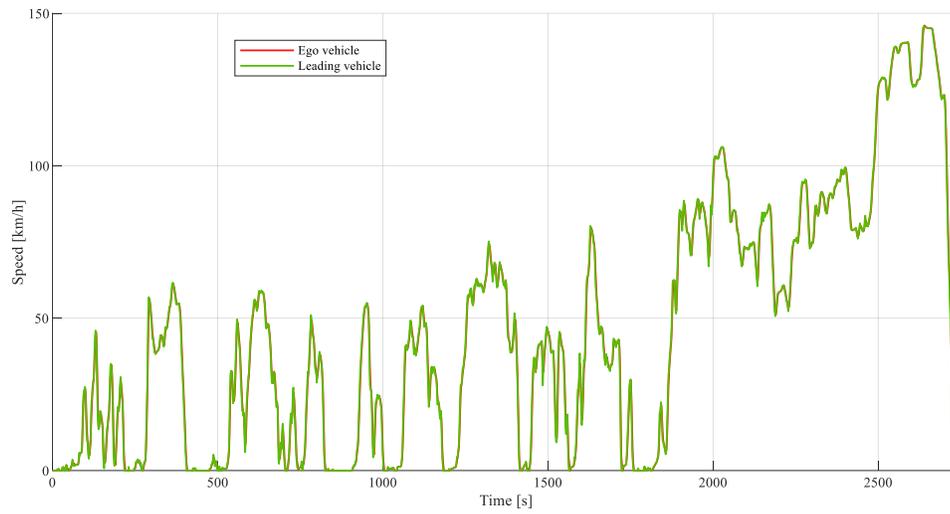


Figure 55 Leading and Ego vehicle speed on Random cycle

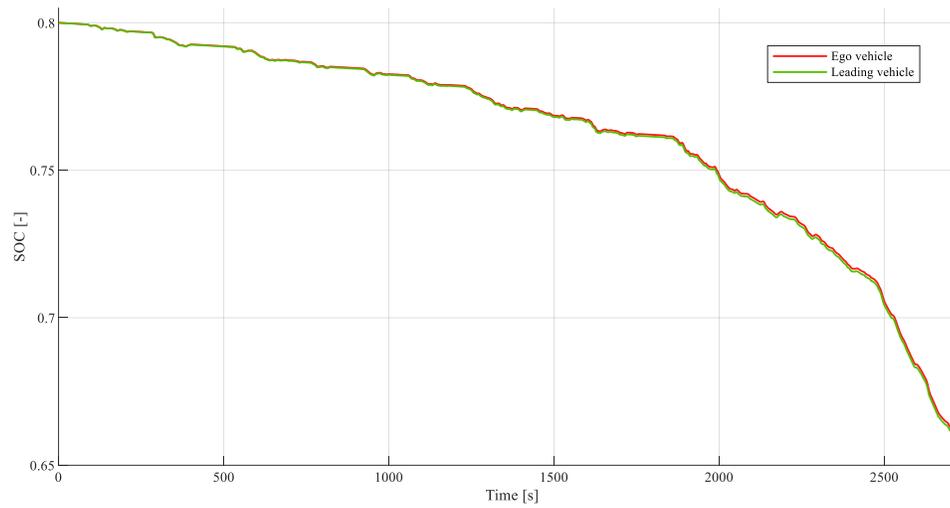


Figure 56 Leading and Ego vehicle SOC on Random cycle

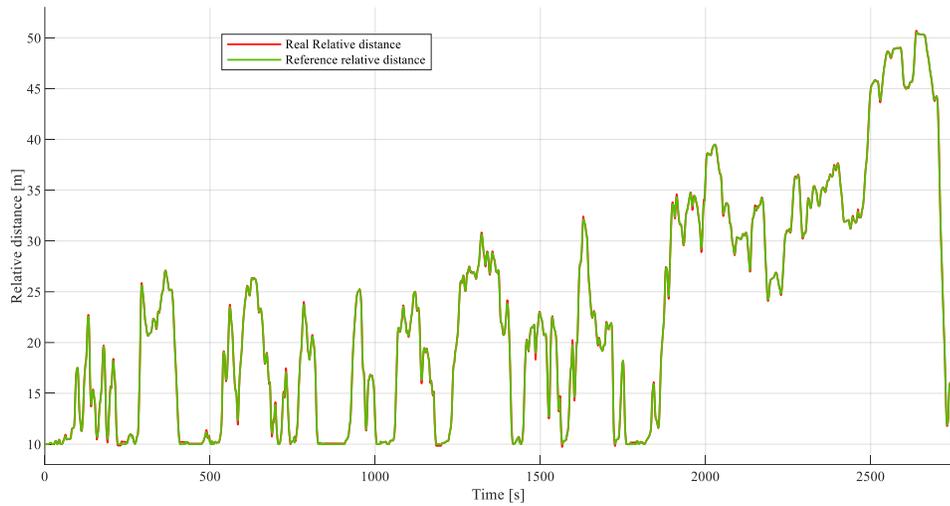


Figure 57 Reference and real relative distance on Random cycle

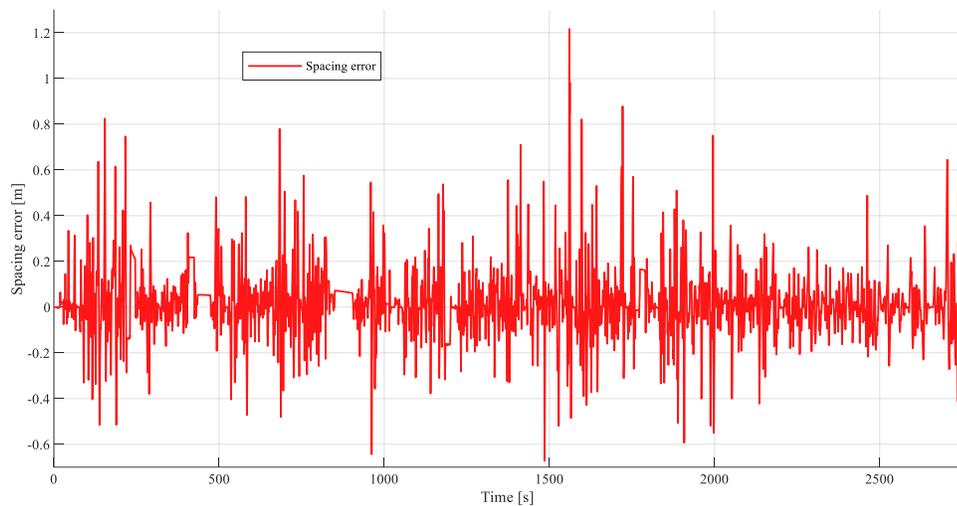


Figure 58 Spacing error on Random cycle

6.1.5 Results analysis

Looking at the results of the previous subparagraphs it is possible to see that the Adaptive MPC works properly and gives the desired results. Indeed, looking at Figure 40, Figure 45, Figure 49 and Figure 55, the ego vehicle speed is very close to that of the leading vehicle on all the drive cycles. This behaviour reflects in a low relative speed, a lot smaller than the required constraints. However, watching Figure 41, Figure 50 and Figure 51 that show a zoomed area of FTP72 and

US06 drive cycles, it is possible to observe that the speed profile of the ego vehicle do not follows all the peaks of high and low speed during the cycle, obtaining a smoother profile. This is one of the main characteristics of MPC controllers, obtained selecting a low value for the weight dedicated to the acceleration increment, and that gives as a results a better driving comfort due to less acceleration and deceleration manoeuvres. These results can be confirmed looking at the acceleration root mean square (RMS) value, that is a parameter used to assess the amount of vibrations during driving. The formulation of the RMS is the following one:

$$RMS_a = \sqrt{\frac{1}{N} \sum_{i=1}^N (a(i))^2} \quad (30)$$

where N is the length of the drive cycle and $a(i)$ is the acceleration at instant i. In Table 5 are displayed the values of RMS for all the drive cycle simulated.

	Leading vehicle RMS [m/s²]	Ego vehicle RMS [m/s²]	Percentage reduction of RMS to the leader
FIP72	0,48	0,42	13,98%
WLTP	0,43	0,39	7,51%
US06	0,79	0,72	8,93%
Random	0,45	0,4	10,29%

Table 5 RMS of acceleration on the different drive cycles

where the percentage reduction of rms to the leader is estimated as:

$$percentage\ reduction\ \% = \frac{RMS_{a,lead} - RMS_{a,followingvehicle}}{RMS_{a,lead}} * 100 \quad (31)$$

As previously said, there is always a reduction of the RMS going from the leader to the follower, obtaining, not only a smoother speed profile, also improvements on State of Charge. In fact, Figure 42, Figure 46, Figure 52 and Figure 56 show that the ego vehicle has a lower energy consumption than the leader, thanks to the less accelerations, but the difference is minimal due the time gap policy, that, when it is employed together with CACC, do not allow an higher

increase on energy consumption. Indeed, the difference on SOC on all the cycle is always lower than 1%, therefore negligible.

At last, looking at Figure 43, Figure 47, Figure 53 and Figure 57, the effective and desired relative distance are shown. The two values are always really close and their difference is smaller than constraints, as it is possible to see in Figure 44, Figure 48, Figure 54 and Figure 58, where spacing error is displayed. This value is the difference between the real and reference relative distance at each instant of the driving cycle and has an absolute value lower than 1,2 on all the cycles.

These results exhibit a correct operation of the controller on several driving conditions. Now a more realistic driving scenario can be tested, to verify the responsiveness and safety of the controller.

7. Four vehicle platoon simulation

The obtained controller has been validated on different drive cycles in a simulation scenario where the platoon is compound only of two vehicles. The leading vehicle follows the selected drive cycle guided by a PI controller, while the succeeding vehicle exploits the MPC controller to implement CACC technology.

Then, numeric simulations are held to certificate the safety of the algorithm in a real traffic scenario. In this study, is considered a four vehicle platoon in which three vehicles are on the same lane and are called leader vehicle, first follower vehicle and second follower vehicle, while another one is following the leading vehicle from the adjacent lane. At a certain instant of the drive cycle this last vehicle performs a cut-in manoeuvre between the leader and the following vehicle, therefore is called cut-in vehicle. The goal of this simulation is to analyse the safety of the controller to the unexpected cut-in manoeuvre. Therefore, it is not considered only if the first following vehicle collides with the cut-in vehicle, also if crashes with the last vehicle of the platoon due to a strong deceleration. Hence, to increase safety and quick reactions, some changes have been made on the controller policy.

First of all the relation on the spacing error, used in the internal model of the Adaptive MPC, is rewritten as follows:

$$\delta = \Delta s_{real,i} * (a - b * e^{-\alpha P}) - \Delta s_{ref,i} \quad (32)$$

where:

- P is a binary parameter, which becomes unitary ten seconds before the cut-in manoeuvre and switches back to zero ten seconds after the overtaking;
- α is a sufficiently large number that nullify the exponential when $P = 1$;
- a and b are two constant value that allow the increase of spacing error.

Using this strategy the cost function will largely increase because the spacing error, computed in the state space model, diverges from its reference value faster and the controller, due to the higher cost function values, and will start to react in advance to the cut-in manoeuvre, ensuring better behaviour to critical events.

It is worth adding that P could result from a cut-in detection technology, however development of this system is out of the scope of this research and some examples can be already found in literature [31, 32].

In addition, to further improve safety, the real relative distance between the leader and the first follower is increased in order to provide an extra distance in case of unpredictable events, such as the overtaking by another vehicle. To obtain this increase in relative distance, the controller reference for the spacing error is shifted from a constant value of zero to a customized signal with an amplitude of 3 meters, a period of 400 s and its width is equal to a quarter of the period. The maximum value of 3 meters is reached with a slope of 40 seconds, while the reference becomes again zero after a negative slope of 15 seconds, as shown in Figure 59.

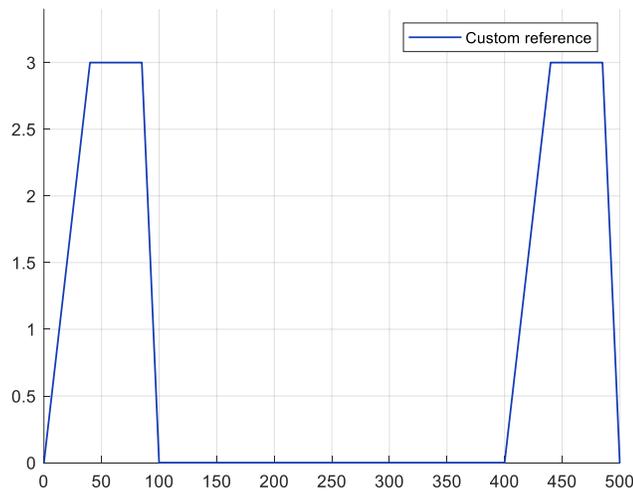


Figure 59 Custom reference of spacing error

These parameters are selected after a sensitivity analysis to support a safe behaviour of the controller without penalizing excessively comfort of vehicle. In fact, since the following vehicle maintains the increased relative distance for 100 s before regaining its original reference, the passengers will not be subjected to continuous acceleration and deceleration to increase and reduce the distance too frequently, but, after a slope of a forty seconds, the result will be only a shift from the reference relative distance profile for 100 seconds. It could also be triggered by advanced sensors as the P value aforementioned or via V2x communications, in case of incoming

vehicles close to the platoon. However, the precise definition of this integration is not brought up in this study and it is simply applied as explained above.

As well as previously done, the comfort increase or reduction can be evaluated considering the RMS value of the acceleration. In Table 6 are shown the results in terms of RMS for two drive cycles, whose overall results will be analysed later.

	US06	US06 (percentage reduction to the leader)	Random	Random (percentage reduction to the leader)
RMS lead	0,787 m/s ²	-	0,446 m/s ²	-
RMS cut-in	0,753 m/s ²	+4,34%	0,420 m/s ²	+5,68%
RMS I follower	0,752 m/s ²	+4,5%	0,465 m/s ²	-4,32%
RMS II follower	0,703 m/s ²	+10,72%	0,443 m/s ²	+0,42% (+4,74% from the I follower)

Table 6 RMS of the acceleration on US06 and Random cycle considering a cut-in manoeuvre

From the results in Table 6 it can be seen that there is a reduction in vibrations going from the preceding to the succeeding vehicle, except for the first following vehicle on Random cycle. Indeed, it is possible to observe the increase of the RMS value for the first following vehicle and this behaviour is caused by accelerations and decelerations resulting from the cut-in manoeuvre and the increase of relative distance, where the second one are particularly important on long-lasting drive cycles. However, the smoother speed profile obtained by the MPC controller allows to soften the growth of RMS for the second follower.

As mentioned above, for the simulation the leader vehicle follows two drive cycles: US06 and a Random cycle obtained by the Random Cycle Generator software. Both the cycles show strong acceleration and deceleration slopes and high speeds, whereby the controller has to be very reactive to perform correctly CACC. During these simulations, the leader is controlled using a PI controller, while the other three vehicles are guided through two different MPC controllers.

The parameters used form the simulation and the controllers are depicted in Table 7.

Parameters	Values	Parameters	Values
T_s	0,1 s	δ_{min}	-5 m
p_1	100 steps	δ_{max}	5 m
m_1	25 steps	Δv_{min}	-10 m/s
p_2	50 steps	Δv_{max}	10 m/s
m_2	15 steps	u_{min}	-3 m/s ²
$d_{0,1}$	10 m	u_{max}	2 m/s ²
$d_{1,2}$	10 m	τ	0,1 s
$d_{0,c}$	5 m	w_δ	1
t_h	1 s	$w_{\Delta v}$	1
$t_{h,1}$	1 s	$w_{\Delta u}$	0,1
$t_{h,c}$	0.5 s	w_{ECR}	100000
a	10	b	9

Table 7 Platoon simulation and controller parameters

where:

- T_s is the sample time, while the fixed-step size is 0.01 s;
- p_1 and m_1 are, respectively, the prediction and the control horizon of the MPC of the first following vehicle;
- p_2 and m_2 are, respectively, the prediction and the control horizon of the MPC of the cut-in and second following vehicle;
- The initial inter-vehicular distance between two vehicles are expressed as: $d_{0,1}$ the distance between the leader and the first follower, $d_{0,c}$ the distance between the leader and the cut-in vehicle and $d_{1,2}$ the distance between the two following vehicles;

- t_h , $t_{h,c}$ and $t_{h,1}$ are the time gap between the leader and first follower, leader and cut-in vehicle and first and second follower, respectively;
- δ_{min} , δ_{max} , Δv_{min} , Δv_{max} , u_{min} and u_{max} are the constraints on spacing error, relative speed and acceleration, respectively;
- τ is the time constant of the engine;
- w_δ , $w_{\Delta v}$, $w_{\Delta u}$ and w_{ECR} are the weights on spacing error, relative speed, acceleration and constraints, respectively;
- a and b are two constant value that allow the increase of spacing error.

The cut-in time instant, chosen for the two cycles and when the overtaking vehicle completes the manoeuvre and becomes the preceding vehicle of the first follower, is $t=473$ s on US06 and $t=2488$ s on the Random cycle. These two instants enable to analyse the reactivity and the safety of the controller in two different and critical phases of the cycles: before a strong deceleration on US06 and during an acceleration ramp at high speed. Figure 60 and Figure 62 show the speed profile of the vehicles on the whole drive cycle, while Figure 61 and Figure 63 focus on cut-in.

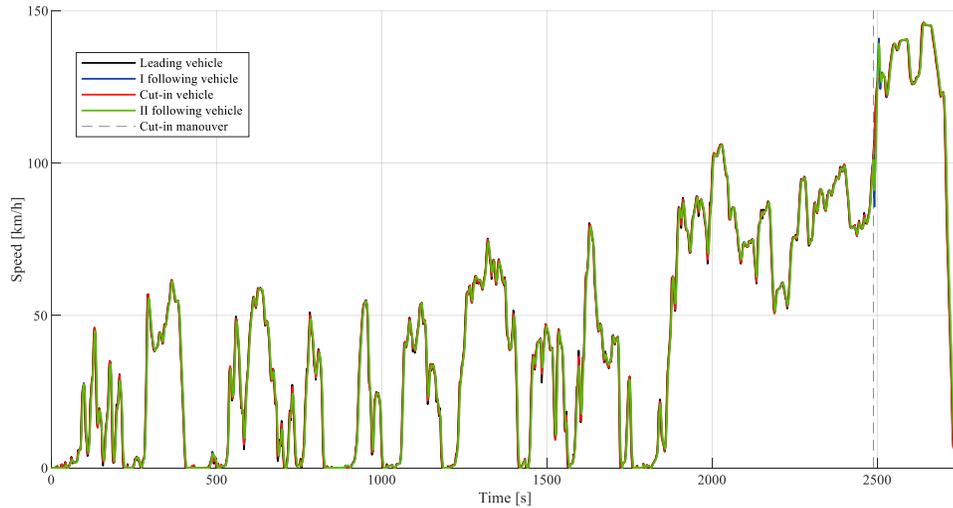


Figure 60 Speed profile of the four vehicles on Random cycle

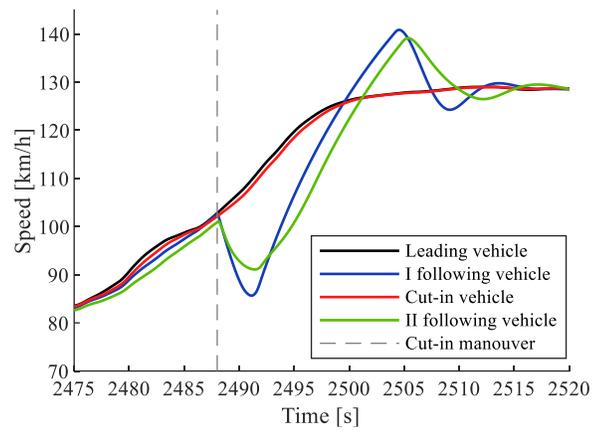


Figure 61 Zoom on vehicles speed over cut-in manoeuvre on Random cycle

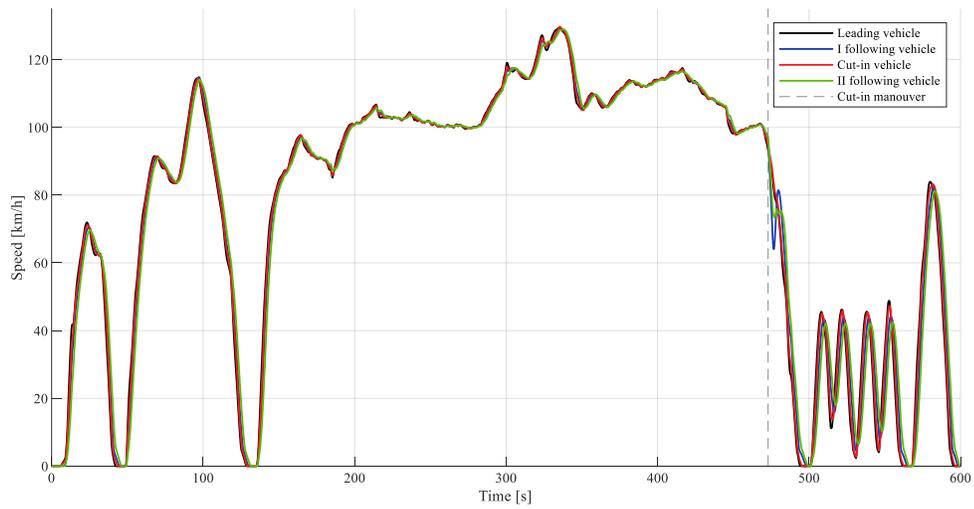


Figure 62 Speed profile of the four vehicles on US06

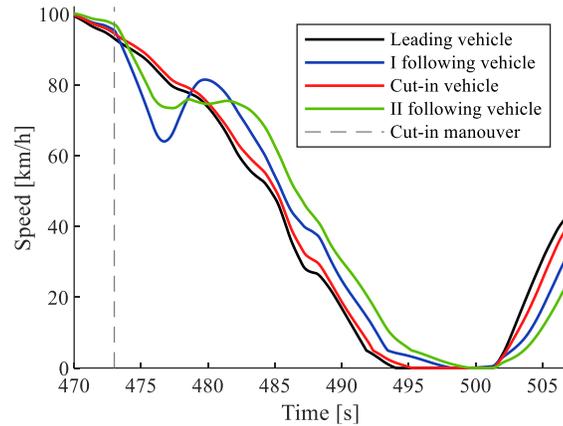


Figure 63 Zoom on vehicles speed over cut-in manoeuvre on US06

The simulation results for the Random cycle scenario can be seen in Figure 60, Figure 61, Figure 64 and Figure 65. It can be seen that the tracking performance on the speed profile are excellent. Therefore, set the relative distance of reference, both the two following vehicles maintain the desiderate gap for the whole cycle without excessive fluctuations on the spacing error. Indeed the oscillations for the second following vehicle lie in a very small range and increase only when the preceding vehicle is overtaken. Also for the first following vehicle small oscillations are obtained and them increase only when the following vehicle gains the extra relative distance. This successful behaviour of the controller is results from the V2V communication that supplies in real time and with high accuracy position, speed and acceleration of the preceding vehicle and from the internal model of the MPC, that computes the state vector all over the prediction horizon and, knowing in advance the future possible actions of the system, determines an optimal control strategy despite the relatively large sample time. Furthermore, the unexpected overtaking, as shown in Figure 65, is handled without critical issues and the desired distance is regained after a transient of 30 s. In fact, the different control strategy adopted for the spacing error over the cut-in manoeuvre, enables a quicker reaction of the controller avoiding collisions in the fleet or an excessive stretch of relative distance.

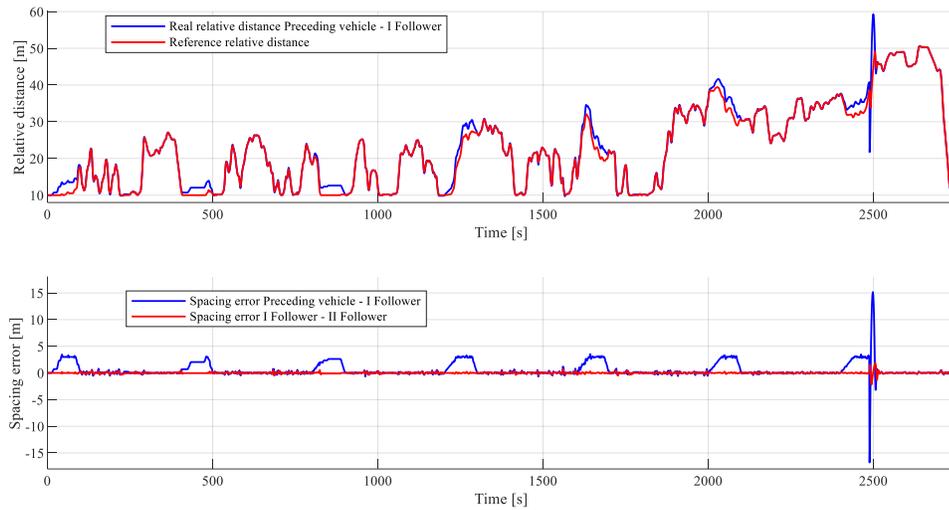


Figure 64 Relative distance and spacing error of the four vehicles on Random cycle

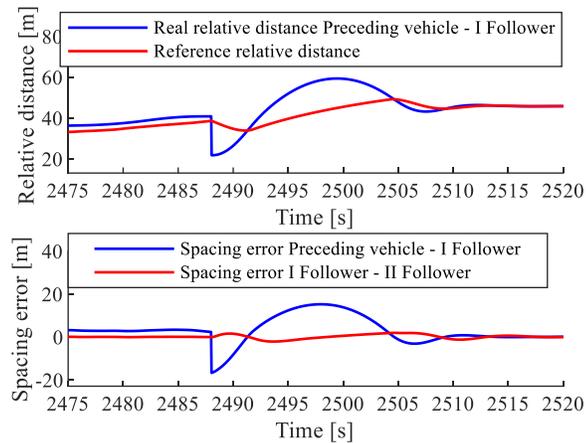


Figure 65 Zoom on vehicles relative distance and spacing error after cut-in manoeuvre on Random cycle

Instead, the control action for the strong deceleration in Figure 63 and Figure 67 is completely different and more difficult to manage. This is a deceleration from 100 km/h to 0 km/h, thus, for this manoeuvre, the gaining of the extra relative distance and the quicker spacing error strategy are essential additions to the control strategy. These expedients avoid collisions to the front vehicle, leading to a minimum relative distance of 10 m at the end of the braking process, adequate to ensure safety. Then, the last 100 seconds of the drive cycle are completed without any other issue. Furthermore, looking carefully at the overtaking area in Figure 63, it is possible to observe that the two following vehicles react immediately to the unexpected event with a

strong deceleration, but, thanks to the smoothing capacity of the MPC controller, the deceleration of -3 m/s^2 is reached quickly only for the first following vehicle, while the second one has a softer deceleration ramp, that improves comfort.

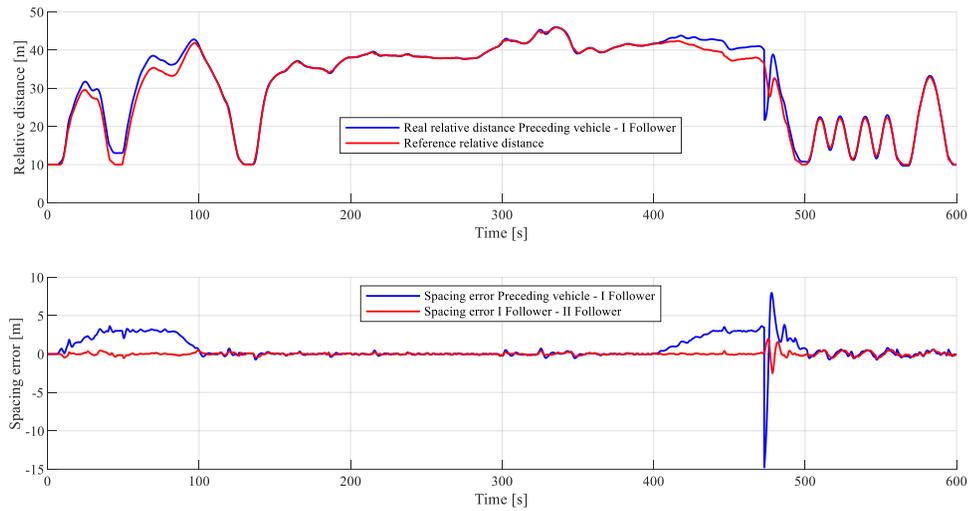


Figure 66 Relative distance and spacing error of the four vehicles on US06 cycle

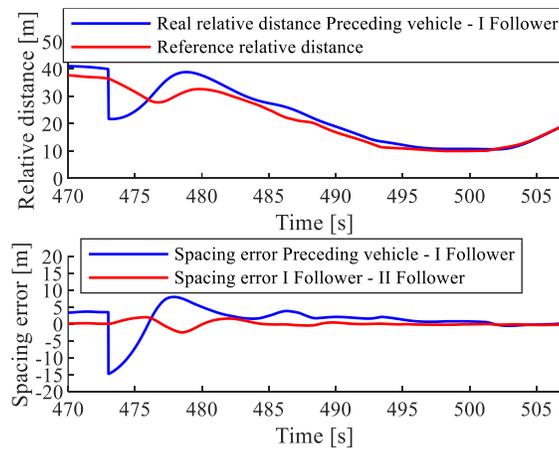


Figure 67 Zoom on vehicles relative distance and spacing error after cut-in manoeuvre on US06 cycle

The results described above show a proper functioning of the controller even in a critical situation as a cut-in manoeuvre, allowing platooning for the autonomous vehicles and guaranteeing a high level of safety without penalizing driving comfort.

8. Conclusions and future works

In this work an Adaptive Model Predictive Control approach is proposed to implement the CACC technology on a battery electric vehicle.

First of all a proper model of the electric vehicle is developed, considering the characteristic of a real passenger car, namely Fiat 500e.

Then, the controller is designed using the MATLAB/Simulink MPC toolbox. However, the shift to an Adaptive Model Predictive Control is required due to the strong nonlinearities of the plant model. The designed controller showed an optimal tracking capacity while ensuring a good drive comfort due to the smoothing of the acceleration profile. The usage of V2V communication was primary in the correct operation of the algorithm, sending in real time information about the position, speed and acceleration of the preceding vehicle.

Afterwards, to prove the safety and the responsiveness of the controller, a simulation considering a platoon of four vehicles, which follow two different drive cycles, was performed. In this scenario a cut-in manoeuvre between the leading vehicle and the first follower was examined. Still during this critical manoeuvre the MPC controller, revised on spacing error and relative distance, showed its ability to respond in quick time and avoid collisions guaranteeing safety and a minimum relative distance of 10 meters, while the driving comfort is kept on high levels and in some cases enhanced.

Future works may be conducted on improving the communication exploring various topological structure, such as Leader Predecessor Follower, or integrating the lateral dynamics to the controller to simulate a more realistic driving scenario.

Figures Index

Figure 1 Model Predictive Control design	6
Figure 2 MPC possible solution	7
Figure 3 Example of control horizon	8
Figure 4 Longitudinal dynamics of the vehicle	12
Figure 5 Equivalent circuit model	14
Figure 6 Internal resistance model	15
Figure 7 BEV model	16
Figure 8 Vehicle body subsystem	17
Figure 9 H-Bridge scheme	19
Figure 10 Battery model block	20
Figure 11 BEV model after changes	24
Figure 12 Efficiency map of the motor	25
Figure 13 Torque and SOC subsystem	25
Figure 14 Brake subsystem	26
Figure 15 Single cell (a) Internal resistance in function of SOC, (b) Open circuit voltage in function of SOC	27
Figure 16 Speed profile of the drive cycle and simulated model	28
Figure 17 State of Charge of the vehicle's battery during the WLTP drive cycle	29
Figure 18 Power required/produced from the motor during the WLTP drive cycle	29
Figure 19 Torque of the motor during the WLTP drive cycle	30
Figure 20 Efficiency of the electric motor in traction during the WLTP drive cycle	30
Figure 21 Efficiency of the electric motor while using regenerative braking during the WLTP drive cycle	31
Figure 22 Battery current during the WLTP drive cycle	32
Figure 23 Battery voltage output during the WLTP drive cycle	33
Figure 24 Define MPC structure and linearize the model	35
Figure 25 Sample time, prediction and control horizon	37
Figure 26 Scale factor dialog box	39
Figure 27 Constraint dialog box	39

Figure 28 Weights dialog box	41
Figure 29 Closed-loop performance slider	44
Figure 30 Adaptive Cruise Control structure in Simulink	45
Figure 31 Speed profile of the leading and following vehicle with ACC on WLTP drive cycle	47
Figure 32 Relative distance between the leading and following vehicle with ACC on WLTP drive cycle	47
Figure 33 SOC profile of the leading and following vehicle with ACC on WLTP drive cycle .	48
Figure 34 Custom MPC Simulink model	49
Figure 35 Speed profile of leading and following vehicle using custom MPC.....	51
Figure 36 Real and reference relative distance using custom MPC	51
Figure 37 Adaptive Model Predictive Control design.....	53
Figure 38 Adaptive MPC block mask.....	54
Figure 39 Simulink function block.....	56
Figure 40 Leading and Ego vehicle speed on FTP72	58
Figure 41 Zoom on vehicles speed between second 345 to 400 on FTP72.....	59
Figure 42 Leading and Ego vehicle SOC on FTP72.....	59
Figure 43 Required and real relative distance on FTP72.....	60
Figure 44 Spacing error on FTP72.....	60
Figure 45 Leading and Ego vehicle speed on WLTP.....	61
Figure 46 Leading and Ego vehicle SOC on WLTP	61
Figure 47 Reference and real relative distance on WLTP	62
Figure 48 Spacing error on WLTP.....	62
Figure 49 Leading and Ego vehicle speed on US06.....	63
Figure 50 Zoom on vehicles speed between second 290 to 350 on US06	63
Figure 51 Zoom on vehicles speed between second 503 to 597 on US06	64
Figure 52 Leading and Ego vehicle SOC on US06	64
Figure 53 Real and reference relative distance on US06.....	65
Figure 54 Spacing error on US06.....	65
Figure 55 Leading and Ego vehicle speed on Random cycle.....	66
Figure 56 Leading and Ego vehicle SOC on Random cycle	66
Figure 57 Reference and real relative distance on Random cycle.....	67

Figure 58 Spacing error on Random cycle.....	67
Figure 59 Custom reference of spacing error.....	71
Figure 60 Speed profile of the four vehicles on Random cycle.....	74
Figure 61 Zoom on vehicles speed over cut-in manoeuvre on Random cycle	75
Figure 62 Speed profile of the four vehicles on US06.....	75
Figure 63 Zoom on vehicles speed over cut-in manoeuvre on US06	76
Figure 64 Relative distance and spacing error of the four vehicles on Random cycle.....	77
Figure 65 Zoom on vehicles relative distance and spacing error after cut-in manoeuvre on Random cycle	77
Figure 66 Relative distance and spacing error of the four vehicles on US06 cycle	78
Figure 67 Zoom on vehicles relative distance and spacing error after cut-in manoeuvre on US06 cycle	78

Table Index

Table 1 Magic Formula Coefficients for Typical Road Conditions	17
Table 2 Fiat 500e parameters	23
Table 3 ACC simulation parameters	46
Table 4 Adaptive MPC parameters	57
Table 5 RMS of acceleration on the different drive cycles.....	68
Table 6 RMS of the acceleration on US06 and Random cycle considering a cut-in manoeuvre	72
Table 7 Platoon simulation and controller parameters	73

Bibliography

- [1] U. o. M. Engineering, "Alberto Bemporad | Embedded Model Predictive Control," 1 August 2016. [Online]. Available: <https://www.youtube.com/watch?v=ugeCx1sytNU&t=3152s>.
- [2] MATLAB, "Understanding Model Predictive Control, Part 2: What is MPC," 30 May 2018. [Online]. Available: <https://www.youtube.com/watch?v=cEWnixjNdzs&t=3s>.
- [3] MATLAB, "Understanding Model Predictive Control, Part 3: MPC Design Parameters," 19 June 2018. [Online]. Available: <https://www.youtube.com/watch?v=dAPRamI6k7Q&t=61s>.
- [4] MATLAB, "Understanding Model Predictive Control, Part 4: Adaptive, Gain-Scheduled and Nonlinear MPC," 27 June 2018. [Online]. Available: <https://www.youtube.com/watch?v=hkYf-Chqwdw&t=1s>.
- [5] Y. Y. J. W. Z. L. J. L. J. N. Fangwu Ma, "Predictive energy-saving optimization based on nonlinear model predictive control for cooperative connected vehicles platoon with V2V communication," *Energy*, vol. 189, 2019.
- [6] T. K. M. M. German Valenzuela, "Nonlinear Model Predictive Control of Battery Electric Vehicle with Slope Information," *IEEE*, 2014.
- [7] D. R. L. a. S. A. Evangelou, "Energy savings from an Eco-Cooperative Adaptive Cruise Control: a BEV platoon investigation," in *18th European Control Conference (ECC)*, Napoli, 2019.
- [8] J. V. E. S. E. S.-K. a. N. v. d. W. Ellen van Nunen, "Robust Model Predictive Cooperative Adaptive Cruise Control Subject to V2V Impairments," in *IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017.

- [9] D. G. X. L. Andreas Weißmann, "Energy-optimal adaptive cruise control combining model predictive control and dynamic programming," *Control Engineering Practice*, vol. 72, 2018.
- [10] H. V. M. D. Seyed Amin Sajadi-Alamdari, "Risk-averse Stochastic Nonlinear Model Predictive Control for Real-time Safety-critical Systems," *IFAC Papers Online*, 2017.
- [11] W. Z. L. W. a. Q. W. Fei Ju, "Iterative Dynamic Programming Based Model Predictive Control of Energy Efficient Cruising for Electric Vehicle with Terrain Preview," *SAE International*, 2020.
- [12] Z. S. Yunli Shao, "OPTIMAL SPEED CONTROL FOR A CONNECTED AND AUTONOMOUS ELECTRIC VEHICLE CONSIDERING BATTERY AGING AND REGENERATIVE BRAKING LIMITS," in *Proceedings of the ASME 2019 Dynamic Systems and Control Conference*, Park City, Utah, USA, 2019.
- [13] MATLAB, "Tire-Road Interaction (Magic Formula)," [Online]. Available: <https://www.mathworks.com/help/physmod/sdl/ref/tireroadinteractionmagicformula.html>.
- [14] F. N. 2. F. 5. Specifications. [Online]. Available: <https://media.fcanorthamerica.com/download.do?id=20137>.
- [15] R. Finesso, D. Misul, E. Spessa and M. Venditti, "Optimal Design of Power-Split HEVs Based on Total Cost of Ownership and CO2 Emission Minimization.," *Energies*, 2018.
- [16] C. D. M. A. M. a. E. S. Maino, "Optimal mesh discretization of the dynamic programming for hybrid electric vehicles," *Applied Energy*, vol. 292, 2021.
- [17] P. a. B. G. Anselma, "Enhancing Energy Saving Opportunities through Rightsizing of a Battery Electric Vehicle Powertrain for Optimal Cooperative Driving," *SAE Intl.*, 2020.
- [18] M. Spano, P. Anselma, A. Musa, D. Misul and G. Belingardi, "Optimal Real-Time Velocity Planner Of A Battery Electric Vehicle In V2V Driving," *IEEE Transportation Electrification Conference*, 2021.

- [19] N. L. a. F. Shladover, "COOPERATIVE ADAPTIVE CRUISE CONTROL (CACC) DEFINITIONS AND OPERATING CONCEPTS," in *TRB 2015 Annual Meeting*, 2014.
- [20] MATLAB, "Design Controller Using MPC Designer," [Online]. Available: <https://www.mathworks.com/help/mpc/gs/introduction.html>.
- [21] M. M. N. L. R. A. Bemporad, "The MPC Simulink Library," 2000.
- [22] MATLAB, "Choose Sample Time and Horizons," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/choosing-sample-time-and-horizons.html>.
- [23] MATLAB, "Specify Scale Factors," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/scale-factors.html>.
- [24] MATLAB, "Specify Constraints," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/specifying-constraints.html>.
- [25] MATLAB, "Tune Weights," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/tuning-weights.html>.
- [26] MATLAB, "Optimization Problem," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/optimization-problem.html>.
- [27] MATLAB, "Adaptive Cruise Control System Using Model Predictive Control," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/adaptive-cruise-control-using-model-predictive-controller.html#d123e33181>.
- [28] MATLAB, "Adaptive MPC," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/adaptive-mpc.html>.
- [29] H. N. M. ., A. T.-S. a. Y. P. F. Hadi Kazemi, "A Learning-Based Stochastic MPC Design for Cooperative Adaptive Cruise Control to Handle Interfering Vehicles," *IEEE TRANSACTIONS ON INTELLIGENT VEHICLES*, vol. 3, no. 3, 2018.

- [30] MATLAB, "Understanding Model Predictive Control, Part 7: Adaptive MPC Design with Simulink," [Online]. Available: <https://www.youtube.com/watch?v=aQhpvrQPxD4&t=16s>.
- [31] "Random Cycle Generator," [Online]. Available: <https://www.tno.nl/en/focus-areas/traffic-transport/roadmaps/sustainable-traffic-and-transport/sustainable-mobility-and-logistics/improving-air-quality-by-monitoring-real-world-emissions/random-cycle-generator/>.
- [32] A. Carvalho, A. Williams, S. Lefèvre and F. Borrelli, "Autonomous Cruise Control with Cut-in Target Vehicle Detection," 2017.
- [33] K. Choi and H. Jung, "Cut-in Vehicle Warning System Exploiting Multiple Rotational Images of SVM Cameras," *Expert Syst. Appl.*, 2019.