Master's Thesis

Double Master's Degree in Industrial Engineering and Automatic Control and Robotics

Torque Vectoring Predictive Control of a Four In-Wheel Motor Drive Electric Vehicle

Author: David Henry González

Director: Dr. Vicenç Puig Cayuela

Call: June 2022

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Abstract

The recent integration of vehicles with electrified powertrains in the automotive sector provides higher energy efficiency, lower pollution levels and increased controllability. These features have led to an increasing interest in the development of Advanced Driver-Assistance Systems (ADAS) that enhance not only the vehicle dynamic behaviour, but also its efficiency and energy consumption.

This master's thesis presents some contributions to the vehicle modeling, parameter estimation, model predictive control and reference generation applied to electric vehicles, paying particular attention to both model and controller validation, leveraging offline simulations and a real-time driving simulator.

The objective of this project is focused on the Nonlinear Model Predictive Controller (NMPC) technique developing torque distribution strategies, specifically Torque Vectoring (TV) for a four-in wheel motor drive electric vehicle. A real-time TV-NMPC algorithm will be implemented, which maximizes the wheels torque usage and distribution to enhance vehicle stability and improve handling capabilities.

In order to develop this control system, throughout this thesis the whole process carried out including the implementation requirements and considerations are described in detail. As the NMPC is a model-based approach, a nonlinear vehicle model is proposed. The vehicle model, the estimated parameters and the controller will be validated through the design of open and closed loop driving maneuvers for offline simulations performed in a simulation plant (VI-CarRealTime) and by means of a real-time driving simulator (VI-Grade Compact Simulator) to test the vehicle performance through various dynamic driving conditions.

Acknowledgements

I would genuinely like to thank Dr. Vicenç Puig Cayuela for his most valuable support and guidance not only throughout the development of this project but also along the double master's journey. He has always been aware of the project progress and the complexities that arose and he has replied to my doubts quickly in every single occasion in a very proactive, helpful and friendly manner. Thank you very much.

I would also like to thank the Vehicle Dynamics team at Applus Idiada, specially, Jaume Cartró Benavides and Javier Gutiérrez Diez for their help, closeness and fellowship at all times. They have always been open to any questions I could ask them and showed great concern in the development of the project.

Last but not least, thanks to my family and friends, who have helped me going through this journey.

Contents

List of Figures

List of Tables

Index of Notation

Acronyms

4WD **F**our-**W**heel **D**rive

ABS **A**nti-lock **B**raking **S**ystem

ADAS **A**dvanced **D**river-**A**ssistance **S**ystems

CRT VI-**C**ar**R**eal**T**ime

DLC **D**ouble-**L**ane **C**hange

DOF **D**egree-**O**f-**F**reedom

ESC **E**lectronic **S**tability **C**ontroller

EV **E**lectric **V**ehicles

HIL **H**ardware-**I**n-the-**L**oop

ICC **I**ntegrated **C**hassis **C**ontrol

IMU **I**nertial **M**easurement **U**nit

IPOPT **I**nterior **Po**int **Opt**imizer

IP **I**nterior **P**oint

ISO **I**nternational **O**rganization for **S**tandardization

LP **L**inear **P**rogramming

LS **L**east-**S**quares

LTV **L**inear **T**ime **V**arying

MF Pacejka **M**agic **F**ormula

MIMO **M**ulti-**I**nput **M**ulti-**O**utput

MIP **M**ixed-**I**nteger **P**rogramming

MPC **M**odel **P**redictive **C**ontroller

NLPDIP **N**on **L**inear **P**rimal-**D**ual **I**nterior-**P**oint

NLSQP **N**on **L**inear **S**equential **Q**uadratic **P**rogramming

NLSQ **N**on **L**inear **L**east-**S**quares

NMPC **N**onlinear Predictive **C**ontroller

PDIP **P**rimal-**D**ual **I**nterior-**P**oint

PeC **Pe**aceful **C**oexistence architecture

PID **P**roportional–**I**ntegral–**D**erivative controller

QCQP **Q**uadratically **C**onstrained **Q**uadratic **P**rogramming

QP **Q**uadratic **P**rogramming

RK4 **R**unge–**K**utta **f**ourth-order method

RPM **R**evolutions **P**er **M**inute

RTI **R**eal-**T**ime **I**teration

SDP **S**emi**D**efinite **P**rogramming

SimWB **SIM**ulation **W**ork**b**ench

SMC **S**liding **M**ode **C**ontrol

sMF **s**implified Pacejka **M**agic **F**ormula

SOCP **S**econd **O**rder **C**one **P**rogramming

SQP **S**equential **Q**uadratic **P**rogramming

TCS **T**raction **C**ontrol **S**ystem

TV **T**orque **V**ectoring

UDP **U**ser **D**atagram **P**rotocol

VTD **V**irtual **T**est **D**rive

Y2F **Y**ALMIP **to F**ORCES Pro

Symbols

α_{ij} slip angle of front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire

Δu input rate vector

δ steering angle

δ_F, δ_i, δ_o front / inner / outer steering angle

$\dot{\psi}$ yaw rate

$\dot{\psi}_{ref}$ yaw rate reference

$\dot{\theta}$ angular acceleration

$\eta_{sim/real}$ simulation time elapsed unit per real time unit

λ hyperparameter

$\mathcal{R}$ turning radius

$\mathcal{B}, \mathcal{C}, \mathcal{D}$ Pacejka coefficients

μ_r, μ_x, μ_y road, longitudinal and lateral friction coefficients

ω angular velocity

ψ yaw angle

ξ_T torque slack variable

a_x a_y longitudinal and lateral accelerations

B wheel brake

C_{ij} cornering stiffness of front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire

d_F, d_R front and real wheel tracks

$e_{T_{req}}$ torque resquest error

F_c^{ij} cornering force in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire (vehicle frame)

F_l^{ij} longitudinal force in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire (vehicle frame)

F_x^{ij} longitudinal force in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire

F_y^{ij} lateral force in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire

F_z^{ij} normal force in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ tire

g gravity constant

H_p prediction horizon

J_z vehicle moment of inertia along z-axis

K control action gain

k time step

K_{us} understeer coefficient

L vehicle wheelbase

L_F, L_R distance from CoG to front / rear axle

m vehicle mass

m_F, m_R front / rear vehicle mass

N horizon length

Q_{e_ξ} torque request error weight

Q_{e_u} input error weight

Q_{e_x} state error weight

$Q_{e_{\tilde{u}}}$ input change error weight

Q_{r_u}	torque rate weight
$Q^*_{r_u}$	tuning variable of Q_{r_u} weight
R	wheel radius
r	input reference vector
R_e	effective wheel radius
R_u	unloaded wheel radius
t	time variable
T_e^{ij}	torque in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ engine
T_s	sampling time
T_{ij}, T_w^{ij}	torque in front/rear $i = \{F, R\}$, left/right $j = \{L, R\}$ wheel
u	control action vector
v	vehicle velocity vector
v_x, v_y	longitudinal and lateral velocities
V_{ch}	characteristic velocity
x	system state vector
X, Y, θ	vehicle pose with respect to inertial reference frame
y	system output vector
β	sideslip angle
β_{ref}	sideslip angle reference
SWA	steering wheel angle
TR	axle-motor transmission ratio

Chapter 1

Introduction

1.1 Motivation

Nowadays, there is a great need to reduce air pollution, global warming, and oil dependency which has motivated not only the use of renewable energies, but also the design of improvements in other areas, such as manufacturing or transportation systems, where the development of electric vehicles (EV) has become fundamental (Parra et al., 2018).

As cars become equipped with more and more features to make them safer and smarter, the complexity of vehicle production has increased, directly affecting energy demand (ACEA, 2021). Also, manufacturers have been working continuously to improve the energy efficiency of production lines. As a result, energy consumption per car produced has decreased by 16.7% over the last 15 years (see Figure 1.1).

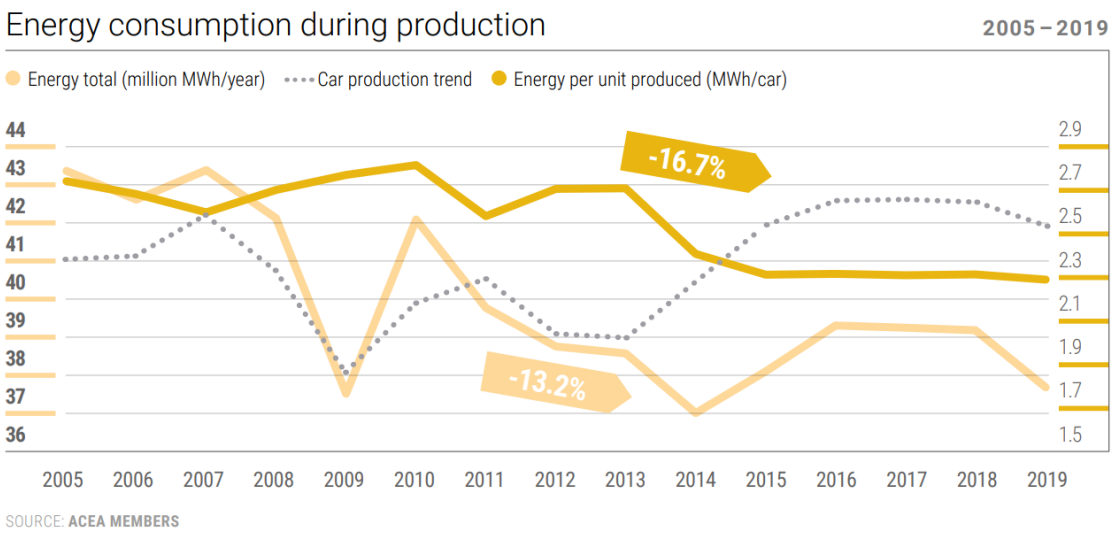


Figure 1.1: Energy consumption during production

Almost 70% of all new cars sold emit less than 130 g CO_2/km which shows a great decrease of emissions from 160 g CO_2/km in the mid 2000s. In addition, the CO_2 emissions per car produced dropped by 37.7% between 2005 and 2019, while the overall figure decreased by 35% over the same period reflecting the industry's efforts to reduce CO_2 emissions from production. The CO_2 evolution is shown in Figure 1.2.

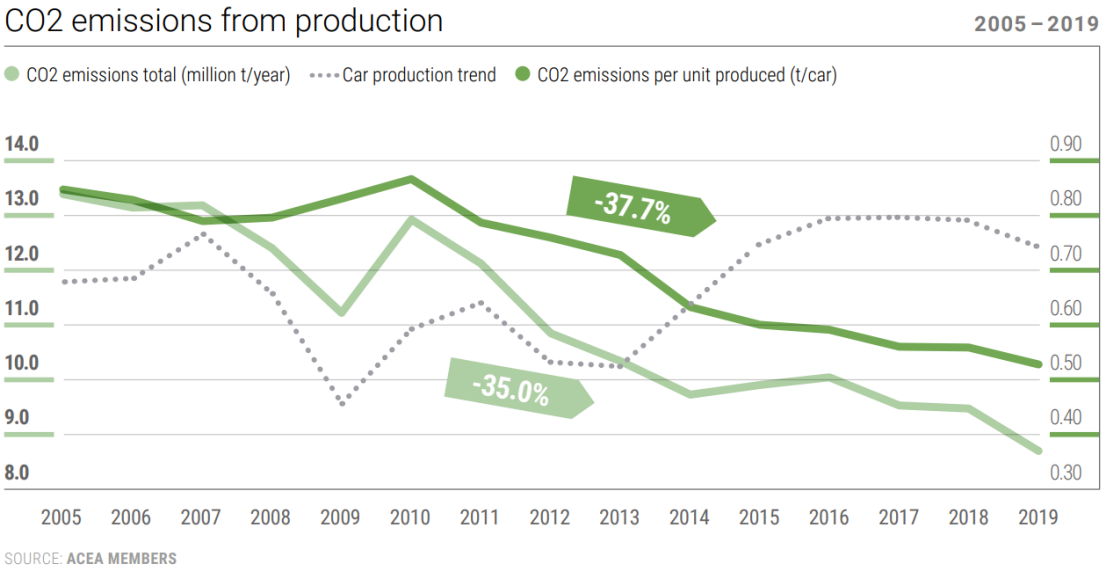


Figure 1.2: CO_2 emissions from production

In the last years, the automotive sector has gained great prominence with the interest in vehicles with electrified powertrains, hybrid and fully electric, becoming one of the main research areas in the automotive industry (Al-Hmouz et al., 2012).

The integration of electric motors in propulsion systems provides higher energy efficiency, lower pollution levels and increased controllability, since electric motors give lower response times (Nikowitz, 2016). These features have led to an increased interest in the development of Advanced Driver-Assistance Systems (ADAS) that enhance not only the vehicle dynamic behaviour, but also its efficiency and energy consumption.

Traditional control approaches have been widely used to implement ADAS during the last decades. However, electrified propulsion systems offer wider complexity than internal combustion propulsion systems as well as more topologies to be chosen. Due to this, modern control or intelligent control approaches have become one of the main research topics in the last years, as they can manage complex systems more easily than traditional approaches.

One of the most complete ADAS systems for enhancing both the dynamic behaviour and stability of an electric vehicle with in-wheel motors is Torque Vectoring (TV) (Inoue

et al., 2016). This technique is focused on the optimal driving torque distribution.

This thesis will be presenting some of the techniques which are important for the development of modern electric vehicles. These techniques have been developed and implemented in both offline simulation and real-time VI-Grade Compact simulator using the BMW 320 dA (2020) vehicle model with several modifications such as using the Tesla Model S (2019) driving torque map to mimic a four-in-wheel motor drive configuration.

The vehicle model is provided by the Vehicle Dynamics group of Applus Idiada and as mentioned further modifications are done to achieve the objective. The work done throughout the thesis, includes vehicle modeling, parameter identification, driving maneuvers design, controller design, vehicle model and controller validation and both offline simulations and real-time machine tests.

Most of the work is focused on the Model Predictive Controller (MPC) approach developing torque distribution strategies since a four-in wheel motor drive electric vehicle has been selected as case study. During the algorithm development, a simulation interface is developed in Simulink for testing each module. Then, in order to carry out a real experimentation, a real-time simulator is used to analyze the vehicle performance under real operating conditions. Section 1.2 proposes detailed objectives of the project and Section 1.3 describes the thesis structure with a detailed description of each topic.

1.2 Objective

The main objective of this project is to deploy a real-time TV-MPC algorithm, which is capable to maximize the wheels torque usage and distribution to enhance vehicle stability and improve handling capabilities. The proposed vehicle model and the online parameters involved will be validated using a simulation plant (VI-CarRealTime) by means of the driver demands (maneuver). Hence, the developed controller also has to be validated in both the simulated scenario and the real-time machine simulator. Based on the principle objective short-term and long-terms goals are formalized:

- Configure vehicle database to be used in VI-CarRealTime and upgrade model with desired configuration (powertrain, wheel-engine scheme, etc).
- Study the dynamics of the vehicle and obtain the parameters of the existing system.
- Design open loop maneuvers for the simulation and track maneuvers for the simulator.
- Validate the vehicle model states with a reference model.

- Design an optimal Nonlinear Model Predictive Controller (NMPC).
- Perform both offline simulations and real experimentation.

1.3 Structure

The document is divided into the following chapters:

1. **Chapter 2: Background.** The chapter is devoted to theoretical studies, recent research and the approach followed.

2. **Chapter 3: Problem Statement.** This chapter describes the problem definition as well as the software used for the developed modules.

3. **Chapter 4: Vehicle Modeling.** In this chapter, the kinematic and dynamic models of the vehicle are detailed. The generation of references is also described.

4. **Chapter 5: Parameter Estimation.** In this chapter, the parameters involved in the vehicle dynamics have to be identified using physical measurements and simulation data.

5. **Chapter 6: Torque Vectoring design.** The real-time control of the electric vehicle applying an NMPC controller is described. The problem formulation as well as the solver used are discussed.

6. **Chapter 7: Implementation.** The implementation details and the subsystems interaction are described. Moreover, both plant model and NMPC will be validated.

7. **Chapter 8: Results.** Open-loop maneuvers results and real simulator results are analyzed in this chapter.

8. **Chapter 9: Environmental, economic and social impact.** This chapter provides an overview of the developed systems impact in the environment, economy and society.

9. **Chapter 10: Budget.** This chapter breaks down the budget to finance the thesis following a cost structure: time dedicated to development and cost of software licenses, among others.

10. **Chapter 11: Conclusions and future work.** Eventually, summarizing conclusions are provided which is essential for further development of this project.

Chapter 2

Background

This chapter presents a literature survey of the development in Torque Vectoring technique in the last years and the approach carried out for this project.

2.1 Torque Vectoring problem

Torque Vectoring is a problem where a vehicle such as a four-wheel drive electric vehicle, varies the wheel torque in a controlled manner to reach optimal vehicle dynamics, that is, the torque distribution is done on each wheel so that it increases both handling and stability among other purposes. Depending on the vehicle features and utility, the aim will be focused on increasing the performance or the comfort and/or stability.

When the vehicle is driving straight, there is a longitudinal weight transfer while accelerating. Rear wheels experience more load than the front ones. Drive torque is equally distributed to the wheels and these rotate at the same speed. When the vehicle decelerates, the behaviour is reproduced just in the opposite manner.

However, torque is only distributed individually along both halfshafts on the axles during cornering. This happens due to the unequal weight distribution on both sides. When cornering, the weight transfer goes from inner to the outer side of the car, leading to loading more the outer wheels. Therefore, when cornering, TV sends more torque to the outer wheels as they have more traction capability and the tire has a higher vertical force. This additional push coming from outer wheels generates a yaw moment which helps to turn the car into the corner. The outer wheels overspeeding causes a yawing effect, without having to brake the inside wheels, which helps boost the cornering capabilities of the car. This effect is depicted in Figure 2.1.

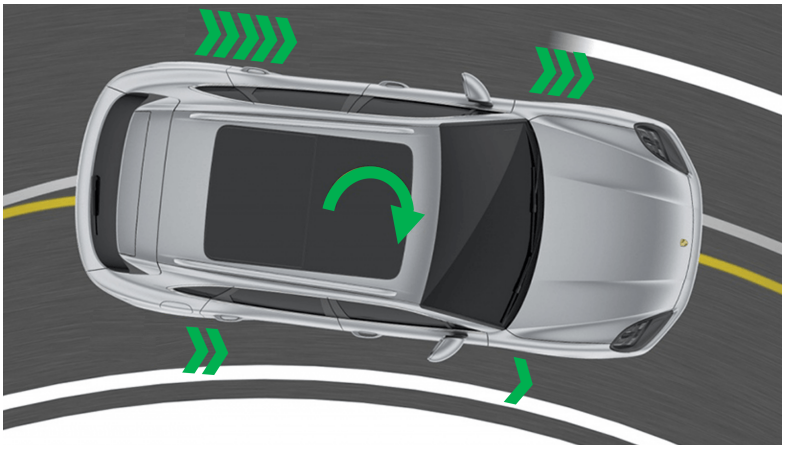


Figure 2.1: TV effect when cornering

Several strategies can be used to address the control the torque distribution in a TV approach. However, most of them are based on controlling the moment along the vertical axis of the vehicle, yaw moment (Lucchini et al., 2020), or directly the yaw rate which is the vehicle's angular velocity about its vertical axis.

Torque distribution approaches have been implemented using a wide variety of control algorithms. Among the traditional approaches, there are simple and advanced algorithms. Regarding the simpler ones, it can be found a proportional-integral-derivative controller (PID) (De Novellis et al., 2014) or a feedforward controller combined with a PID. In adittion, more sophisticated controls have been proposed to address this problem such as Sliding Mode Control (SMC) (Haddoun et al., 2006) (or second-order Sliding Mode Control, SO-SMC) and MPC (Metzler et al., 2020). The latter provide enhanced results, although their computational cost is higher than the conventional PID approaches.

In recent years, intelligent methods have become more important. Neural networks (Haddoun et al., 2006) or fuzzy logic systems (Jalali et al., 2009), or even a neuro-fuzzy approach (Parra et al., 2018) can provide good results with lower computational cost although finding an appropriate number of layers and membership functions can be both very challenging and time consuming.

Once a general overview of the methodologies of torque distribution have been covered, most relevant research for the conceptual idea that is behind the thesis will be discussed.

In (Kang et al., 2012), a high-level controller for Four-Wheel Drive (4WD) EV consisting of three parts (a supervisory controller, an upper-level controller, and an optimal torque vectoring algorithm) is proposed. This controller computes the traction force and yaw moment inputs to track the desired dynamics and leverages an optimal torque vectoring algorithm to determine actuator commands.

Along with the control algorithm used, there is a real-time requirement when the control is designed not only to perform simulations but to be tested on a real vehicle. To monitor and actuate over a vehicle, a constrained optimal control approach must handle real-time constraints (Kasinathan et al., 2016). Thus, any computation and the control action must be done before the specified time expires. If not, a loss of control may lead to a dangerous and an undesired behaviour.

In the scope of active safety systems dealing with real-time constraints, most solutions are focused in the control of the vehicle lateral dynamics by means of either implicit (conventional) or explicit linear MPC formulations according to the problem dimensionality and online computation capacity. In (Barbarisi et al., 2009), a yaw stability controller based on a Linear Time Varying MPC (LTV-MPC) formulation and on a two-track vehicle model using independent braking of the four wheels is presented.

Nonlinear MPC (NMPC) are proposed in the literature employing the real-time iteration (RTI) scheme originally proposed in (Diehl et al., 2002b) on an autonomous vehicle obstacle avoidance application using a nonlinear four-wheel vehicle model and a nonlinear tire model to derive the track-dependent (spatial) dynamics (Frasch et al., 2013). Moreover, in (Siampis et al., 2017) an NMPC strategy for the stabilization of a vehicle near the lateral acceleration limit taking advantage of the rear axle electric torque vectoring configuration is proposed. A nonlinear four-wheel vehicle model that neglects the wheel dynamics is combined with a nonlinear tire model is considered.

A recently growing literature discusses the energy-efficient TV using multiple powertains. In (Parra et al., 2020), a NMPC implementation for energy-efficient TV based on the reference yaw rate and wheel torque allocation is proposed. The cost function weights are modified through fuzzy logic that adaptively prioritize vehicle dynamics or energy efficiency, depending on the driving conditions.

2.2 Proposed Approach

TV is a well established strategy, addressed in the literature in different ways, but it is more focused on the simulation of small scale vehicles rather than commercial ones. High-computational demand controllers were not suitable due to the computation limitations of conventional automotive Electronic Control Units (ECUs). However, as time goes by, the computational load that these units handle increases due to the adding of high demanding ADAS in the production vehicles. So, a proper study of a real-time optimal MPC applied to a real scale vehicle can be very promising to improve vehicle dynamics. The complete description of the proposed approach taken is presented in Chapter 3.

Chapter 3

Problem Statement

In this Chapter, the TV problem structure will be initially introduced in a global manner allowing the reader to understand the context of the problem and later, the specific approach to be addressed will be well defined. But before entering into further details the MPC algorithm must be presented.

3.1 Model Predictive Control (MPC)

MPC uses a plant model to predict its future behavior by solving an online optimization algorithm that allows to select the best control action so that the predicted output matches the reference.

A key aspect of an MPC is that it is a multivariable controller that controls the outputs simultaneously by taking into account all the interactions between system variables. So, it can handle multi-input multi-output (MIMO) systems. Likewise, it can handle constraints either physical, operating or safety limitations.

Another important feature is its preview capability, which is similar to feedforward control. In addition, an MPC can easily incorporate future reference information into the control problem to improve controller performance.

Once the controller capabilities have been described, the working principle will be detailed. The control scheme can be seen in Figure 3.1 where MPC controller includes the optimizer and the plant model. The reference, r, the control action, u and the plant output, y are also depicted. Note that, as MPC handle MIMO systems the previous signals are vectors.

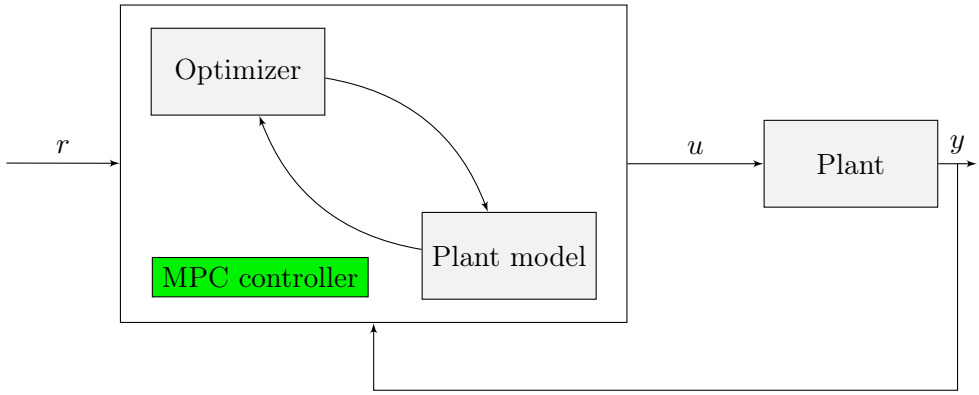


Figure 3.1: MPC control scheme

The MPC follows a receding horizon control principle. Rather than optimizing for the full trajectory into the future, the controller starts from the current time step considering only the current states and then from there a look ahead is taken. Therefore, a limited preview into the future is considered (Peters, 2021). The number of future steps is called prediction horizon, H_p. So, the state is measured and the optimizer is tasked to optimize the next time steps into the future. An optimal solution, sequence of inputs that brings the states to the reference, that minimizes the objective function is obtained for the given number of time steps. Only the very first optimized input will be applied to the system and then, the new state is measured and a re-plan is done from there. The idea is rather than playing out these inputs in open-loop, only the first input is applied and later, measuring again where the system has evolved. The MPC prediction diagram can be seen in Figure 3.2.

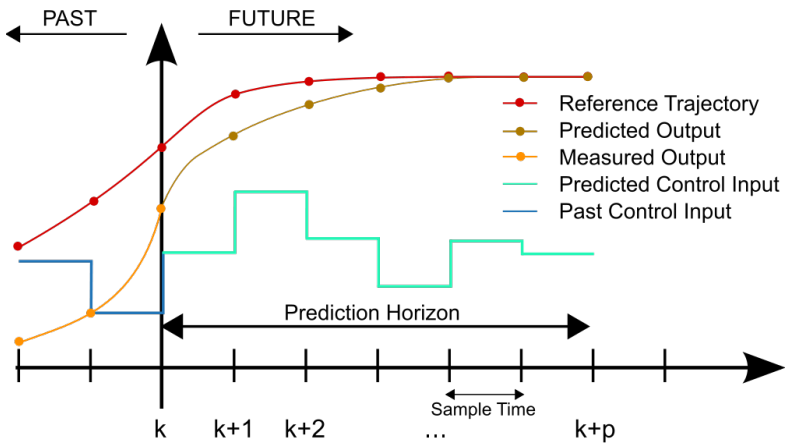


Figure 3.2: MPC prediction diagram

After applying the first input, the system will reach a given value that may not be exactly where it was planned but the difference will be negligible if a reasonably accurate plant model is chosen. Now, the system variables are measured and afterwards, the re-

plan is done from there for the next H_p in order to get the next trajectory. The MPC is able to sort of capture these errors because we always crave the system for the true state. In this way, the controller is also able to account for errors by re-planning. Feedback is introduced, and the problem size is reduced, since planning it is only needed for the next time steps but not for the whole trajectory.

3.2 TV general scheme

To facilitate the understanding of the general control scheme of a TV problem, a high-level Integrated Chassis Control (ICC) scheme (Mazzilli et al., 2021) is shown in Figure 3.3 in which special emphasis is given to the elements that intervene in the system (Diez et al., 2018). An integrated vehicle control system is an advanced system that coordinates all the chassis control systems and components to improve the overall vehicle performance including handling stability, ride comfort, and safety, through creating synergies in the use of sensor information, hardware, and control strategies of different control systems.

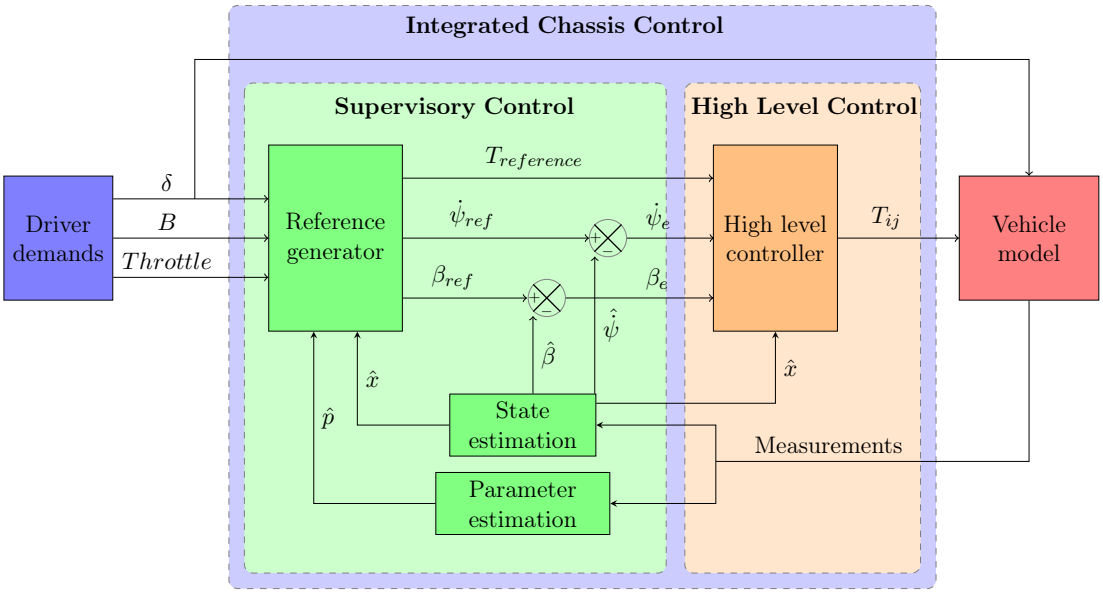


Figure 3.3: ICC control scheme

The main adopted framework in the literature is focused on a centralized control scheme which leverages both captured data by sensors and provided by the driver demands as inputs (steering angle, δ; brakes, B; and throttle). The first layer found in the ICC scheme is the Supervisory Control, which includes: a reference generator providing the yaw rate and sideslip references ($\dot{\psi}_{ref}, \beta_{ref}$) and the torque reference; a state estimator; and a parameter estimation, the latter ones based on the available measurements. Moreover, state errors are computed to feed the next layer as well as states.

The High Level Control layer can gather several high-level control systems and additionally Control Allocation module. The former uses the vehicle motion demands, i.e. yaw rate reference, to compute the motor torques on each wheel in order to simultaneously deliver the longitudinal total force request and the vehicle yaw moment demand. The torque request can be provided by either the Supervisory control layer or the driver, or by the High-level Control layer. The only difference lies in the use of engine operating curves, as a throttle/torque map, and a computation from the yaw rate demand, respectively.

These layers end-up with the computation of the wheel torque that will be applied to the vehicle. The vehicle also receives the driver steering demand signal. Once the loop finishes, vehicle variables are measured and estimated, and the procedure is repeated until the vehicle is no longer used (turned off).

ICC framework addresses several control problems: TV, Electronic Stability Controller (ESC), Anti-lock Braking System (ABS), Traction Control System (TCS) among others. In this case, only the TV control is shown but it can be easily extended to incorporate more controllers within the high-level control scheme. The main benefit lies on moving from a peaceful coexistence architecture (PeC), in which each chassis actuation controller is tuned separately, to a scenario where all controllers are regulated by a coordination layer that acts as a brain.

3.3 Main Contribution and Outline

The complete design, integration and validation of an ICC problem are beyond the scope of the thesis. Therefore, it has no sense to cover and model all systems due to their great complexity and size, but rather to focus on the most interesting aspects of the TV approach.

In order to present the addressed systems within the ICC strategy, a TV control scheme is proposed in Figure 3.4. The hypothesis and considerations taken into account will be described afterwards.

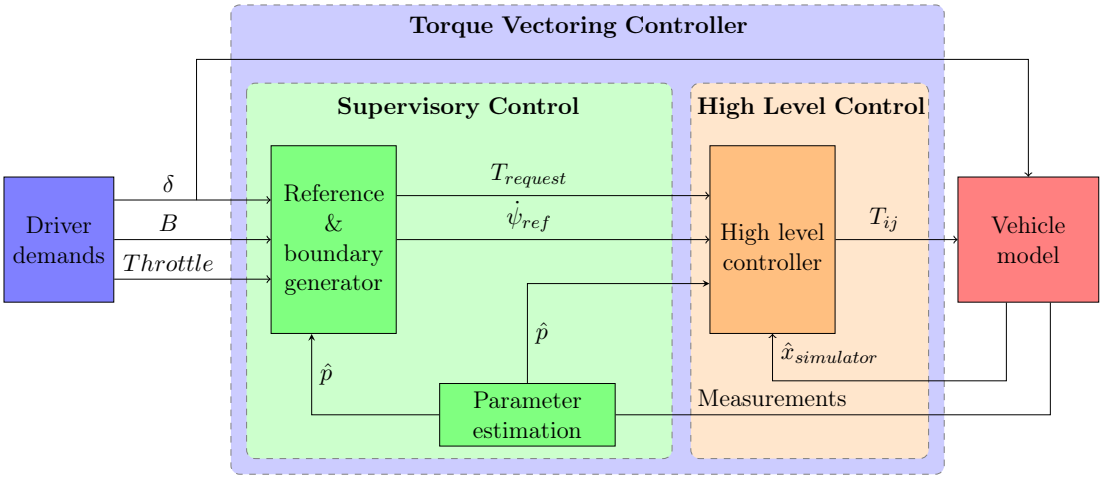


Figure 3.4: TV control scheme

First of all, as the project will be developed using either an offline or a real-time simulator the driver demands will be provided by the simulator by means of the execution of a driving maneuver and by the driver interactions with the vehicle (steering, pedals, etc), respectively.

The Supervisory Control layer includes: a reference and boundary generator which computes the total torque request from the motor pedal map using the wheel/engine speed and the throttle (%) signals and the yaw rate reference; and a parameter estimation module which identifies system parameters and compute both offline parameters at each control iteration and online parameters at each plant iteration. Therefore, the parameters that remain constant and those that vary over time are estimated and leveraged to generate the references and to be used in the controller.

The estimation of the states is not addressed in this case. In any real case scenario, such as a control setup in a real car, this module is fundamental. States have been considered to be measurable as a simulator is used to carry out the experimentation. The load and time it takes to design an appropriate state observer is devoted to the development of other systems that are quite crucial for the vehicle stability and performance. In fact, as the NMPC is a model based approach, an accurate vehicle model fitting will be great for not being far from the expected behaviour.

If a real chassis platform were available, it would be necessary to capture the vehicle states. Therefore, the information given by sensors, (i.e. gyroscope, accelerometer, inertial measurement unit (IMU), engine current) should be processed and integrated.

The High Level Control layer incorporates the TV-NMPC module which computes the optimal wheel torques that will be applied to the vehicle.

3.3.1 TV specifications

With the overview of the TV control scheme outlined, it is time to delve into the controller specifications and the hypotheses that have been assumed.

The vehicle model is a four-wheel double-track nonlinear model, whose front wheels can steer an angle, δ, without rear-steering capabilities, which can be seen as a next step to be integrated in the ICC logic. Lateral, longitudinal and vertical dynamics will be considered to improve model fidelity. Regarding the tire forces, the longitudinal tire forces are directly computed from the torque applied to the tires through the axle-motor transmission ratio; the Simplified Pacejka model (Magic Formula) will be used to model the tire lateral forces; and the normal tire forces are obtained considering the contribution of: the static vertical load in each tire and the load transfer associated with both longitudinal and lateral acceleration in steady-state cornering, including the tangential acceleration and the sideslip rate.

The computation of the tire sideslip angles, assumes small angles and considers the track width in each axle. Then, for simplicity, pure lateral and pure longitudinal slip conditions will be considered instead of combined slip conditions.

The state variables are the sideslip angle, β and the yaw rate, $\dot{\psi}$ and the control variables are the wheel torques, T_{ij} being i the front or rear axle and j the left or right wheels. The controller variables are shown in Equation (3.1). In Figure 3.5, the control variables, x, and manipulated variables, u, are shown within the four-wheel vehicle model

$$x = \begin{bmatrix} \beta \\ \dot{\psi} \end{bmatrix}, \qquad u = \begin{bmatrix} T_{FL} \\ T_{FR} \\ T_{RL} \\ T_{RR} \end{bmatrix} \tag{3.1}$$

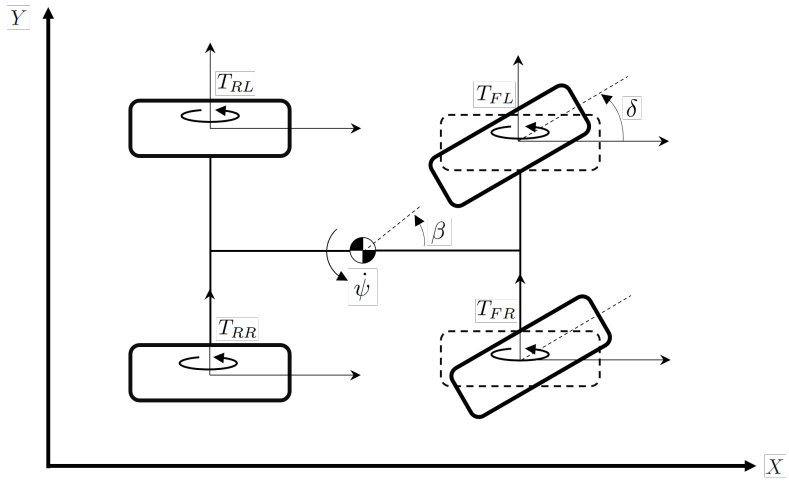


Figure 3.5: Control and manipulated variables

Regarding the controller, the NMPC strategy works with a frequency of 100 Hz while the plant works with 1 kHz, meaning that it updates the system variables 10 times faster than the controller. The prediction horizon is not large due to the nature of the problem to solve and the real-time constraint. The TV is continuously acting over time not to reach the limit handling capabilities. Therefore, it is not necessary to have a large prediction horizon in our system.

As far as the objective function is concerned, it must be said that it includes five cost terms to minimize: the yaw rate error; the change in input (smooth behaviour) throughout control executions; the torque error; the torque request error (difference between the sum of all wheel torques and the request torque at each time step); and the error of the current input with respect to the optimized one of the previous optimizer call in order not to have input noticeable offsets over time. Several constraints have been used to leverage the system knowledge to improve handling capabilities.

Throughout the development of this thesis, the different topics addressed will be described in detail as well as the implementation and integration of subsystems in Simulink with the aim of building a generalized control scheme that in addition to fulfill the performance specifications satisfies the real-time constraint.

Chapter 4

Vehicle Modeling

In order to correctly estimate parameters and carry out control techniques, the use of a vehicle model is crucial regardless of the fact of having a vehicle model simulator. This model must encapsulate most of the dynamics displayed in the vehicle, which implies a trade-off between the model complexity or fidelity and the computational demand. Note that when a controller is designed for real-time applications, there exists a restriction related to computation efficiency which leads to a reduced computational load model which can describe the essential dynamics of a vehicle.

The model is built according to the dynamic and kinematic equations given by a four-wheel vehicle. The kinematic model provides a simple mathematical description of the vehicle movement. The motion equations that govern the system are based on the geometrical relationships. Several options can be found in the literature to carry out the mathematical representation such as the classical bicycle (single-track) model (Rajamani, 2011) (see Figure 4.1). This model is mainly used for estimating the vehicle lateral dynamics by considering both lateral and yaw motions and neglecting longitudinal, pitch, roll, and vertical motions.

However, there are many other options to model the vehicle lateral motion with a higher level of complexity. These models are often integrated dynamical models that consider both vehicle longitudinal and lateral velocities simultaneously, by observing the tire slip angle.

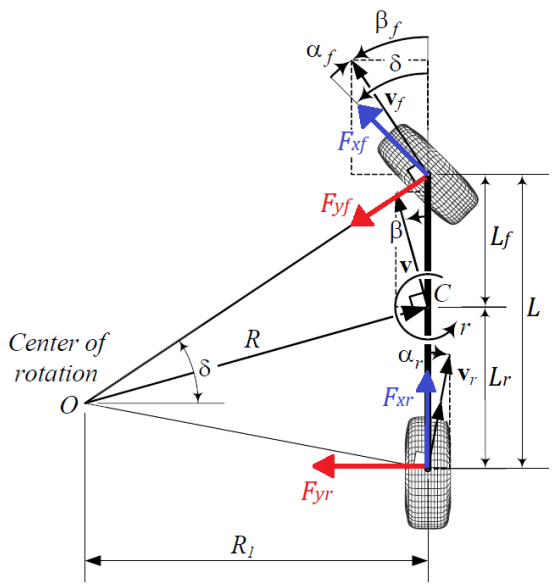


Figure 4.1: Single-track vehicle model

Single-track models can explain a large part of dynamic behaviour during handling of the vehicle, especially during stationary ride but these models do not take into account the redistribution of weight in the lateral direction in dynamic conditions, nor the angular oscillations of the vehicle around the longitudinal axis (Miloradovic et al., 2019). For this reason, the spatial two-track models of the vehicle are used instead, which observe the behaviour of each tire separately and which may be used in the design of the electronic stability control systems of the vehicle. Thus, in this thesis, the double-track model (Jin et al., 2019) will be taken into account.

The double-track model assumes that the vehicle movement is planar motion. Moreover, pitch, roll, and vertical vehicle motions are neglected. When the longitudinal motion is added and the dynamics for four wheels are also addressed for the single-track model, the four-wheel vehicle model involves longitudinal and lateral motions and yaw motion leading to the so-called double-track model.

4.1 Kinematic Model

Before describing the kinematic and dynamic equations, reference frames must be addressed. The inertial frame is denoted as (X, Y) axis and the vehicle frame as (x, y) axis. The frames chosen are orthonormal, right-handed with the z-axis pointing up, and the y-axis pointing left. The kinematics and motion equations are written with respect to the vehicle's center of gravity (CoG). In addition, the CoG is assumed to be located halfway between the left and right side of the car.

The kinematic geometry model is shown in Figure 4.2. The relation between the velocity of the vehicle with respect to the inertial frame is derived using vehicle longitudinal velocity, v_x; lateral velocity, v_y; and angular velocity, ω.

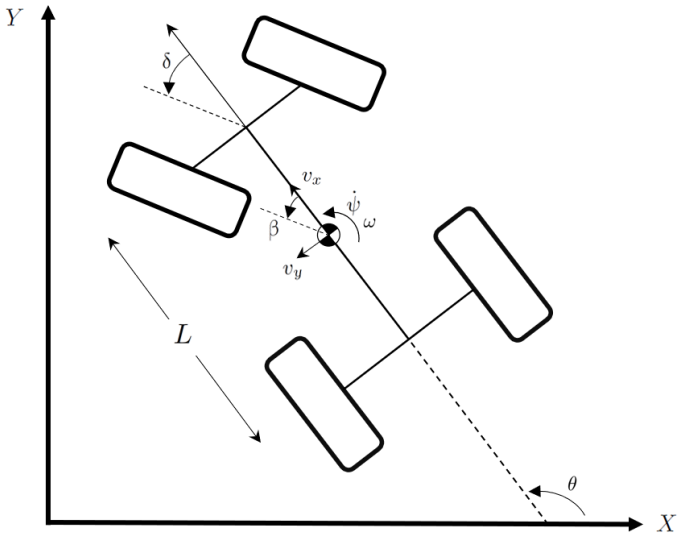


Figure 4.2: Kinematic geometry model

The kinematic equations are derived from the free-body diagram yielding to

$$\dot{X} = v_x \cdot cos(\theta) - v_y \cdot sin(\theta) \tag{4.1}$$

$$\dot{Y} = v_x \cdot sin(\theta) + v_y \cdot cos(\theta) \tag{4.2}$$

$$\dot{\theta} \equiv \dot{\psi} = \omega \tag{4.3}$$

where $\dot{X}$, $\dot{Y}$ are the longitudinal and lateral velocities, respectively; $\dot{\theta}$ is the angular acceleration; ψ is the yaw rate; and v_x, v_y are the longitudinal and lateral velocities.

4.2 Dynamic Model

The double-track vehicle model is shown in Figure 4.3. Note that the longitudinal and cornering tire forces must be transformed to the vehicle frame in the front tires since in the rear ones, both cornering and longitudinal forces match the vehicle lateral and longitudinal forces. Thus, the vehicle forces in the inertial frame can be written as

$$F_x^{FL} = F_l^{FL} \cdot cos(\delta) - F_c^{FL} \cdot sin(\delta) \tag{4.4}$$

$$F_y^{FL} = F_l^{FL} \cdot sin(\delta) + F_c^{FL} \cdot cos(\delta) \tag{4.5}$$

$$F_x^{FR} = F_l^{FR} \cdot cos(\delta) - F_c^{FR} \cdot sin(\delta) \tag{4.6}$$

$$F_y^{FR} = F_l^{FR} \cdot sin(\delta) + F_c^{FR} \cdot cos(\delta) \tag{4.7}$$

where δ is steering angle at front wheels and F_k^{ij} correspond to the longitudinal/lateral force, $k = \{x, y\}$, applied in the front/rear axles, $i = \{F, R\}$, and left/right sides, $j = \{L, R\}$.

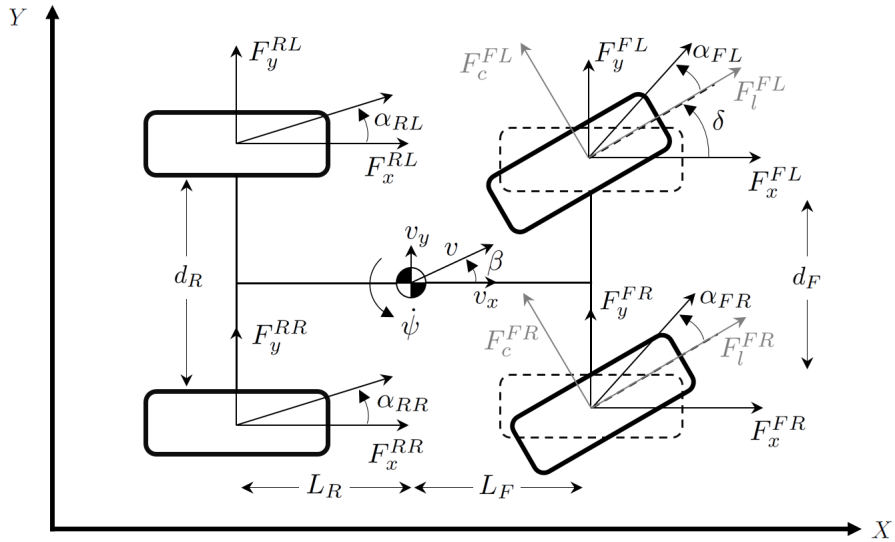


Figure 4.3: Double-track vehicle model in inertial coordinates

The prediction model with yaw rate, $\dot{\psi}$, and sideslip angle, β, will be used as the internal plant of the NMPC and it is given by the lateral force and yaw moment balance equations (Metzler et al., 2020)

$$\begin{aligned}\dot{\beta} = \frac{1}{mv_x}\Big[&(F_x^{FL} + F_x^{FR}) \cdot sin(\delta - \beta) + (F_y^{FL} + F_y^{FR}) \cdot cos(\delta - \beta) \\ &-(F_x^{RL} + F_x^{RR}) \cdot sin(\beta) + (F_y^{RR} + F_y^{RR}) \cdot cos(\beta)\Big] - \dot{\psi}\end{aligned} \tag{4.8}$$

$$\begin{aligned}\ddot{\psi} = \frac{1}{J_z}\Big[&(F_x^{FL} + F_x^{FR}) \cdot sin(\delta) \cdot L_F + (F_y^{FL} + F_y^{FR}) \cdot cos(\delta) \cdot L_F \\ &-(F_x^{RL} + F_x^{RR}) \cdot L_R - (F_y^{FR} - F_y^{FR}) \cdot cos(\delta) \cdot \frac{d_F}{2} \\ &+(F_x^{FL} - F_x^{FR}) \cdot sin(\delta) \cdot \frac{d_F}{2} - (F_y^{RL} - F_y^{RR}) \cdot \frac{d_R}{2}\Big]\end{aligned} \tag{4.9}$$

where d_F, d_R are the front and rear wheel tracks, respectively, and L_F, L_R are the front

and rear semi-wheelbase which is the distance from the CoG to each axle.

4.2.1 Simulation plant

The simulation plant works with a frequency of 1000 Hz, that is, the sampling time, $T_s = 1\,ms$. Typically, values of the order of milliseconds are used when the plant to be controlled has fast dynamics. On the other hand, when the dynamics of the system is slow, the sampling time can be increased up to the order of minutes.

4.2.1.1 VI-CarRealTime plant

Firstly, a vehicle simulator will be introduced. VI-CarRealTime is a virtual modeling and simulation environment targeted to a simplified four-wheel vehicle model. This tool include the ability to assemble the vehicle system by collecting its fundamental subsystems, specifying dynamic maneuver schedules, launching standalone or Matlab-Simulink embedded simulations, etc (VI-GRADE, 2021).

The environment consists of:

- Symbolically derived parameterized equations of motion
- Pacejka tire model
- Virtual driver model

All system data could either come from experimental tests performed in a lab, or from a virtual test performed within Adams Car.

Delving into the vehicle model, VI-CarRealTime contains a simplified model of a four-wheeled vehicle. The vehicle is a reduced degree-of-freedom (DOF) model. It includes only 14 degrees of freedom distributed as follows (see Figure 4.4).

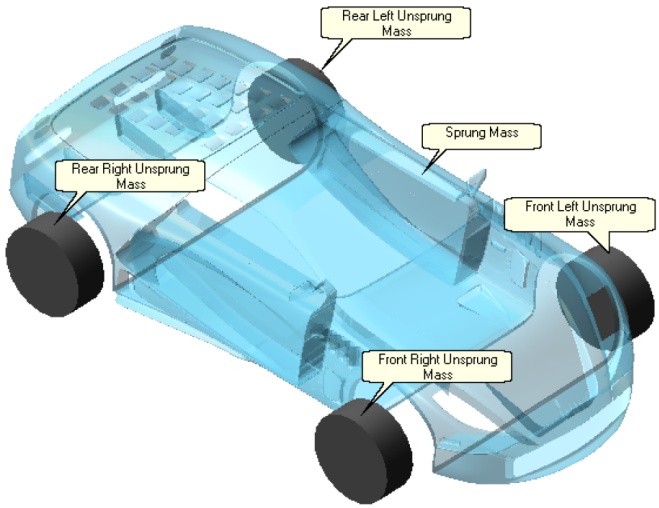


Figure 4.4: CarRealTime vehicle model

The model includes five rigid parts:

- vehicle chassis (sprung mass)
- four-wheels parts (unsprung masses)

The vehicle chassis has 6 DOFs while wheel parts have 2 DOFs each (one for describing the motion with respect to the vehicle body and the other wheel spin).

The VI-CarRealTime suspension does not have linkages or bushings, and the steering system does not have parts for the steering wheel or rack. Suspension and steering system properties (kinematic, compliance and component data) are described by lookup tables using a conceptual approach. Other vehicle subsystems (brakes and powertrain) are described using differential and algebraic equations.

The vehicle model's intent is to accurately predict overall vehicle behaviour for cornering, braking, and acceleration-performance studies for four-wheeled vehicles with independent-front and independent-rear suspensions.

The simplified model is described in terms of commands and functions that use an internal development environment working as a symbolic manipulator tailored for deriving multibody equations and a code generator. Given the model description, it outputs C code for a simulation program for the particular model that is described. Parameters for the model, i.e wheelbase, spring stiffness can be changed at run time and are passed in by input files.

The simplified vehicle runs faster than real time, depending on the computer platform. This makes the model potentially useful for Hardware in the loop (HIL) simulations and

driving simulators. Because of the fast simulation times, the model is also useful as a plant for controller design, and to do optimization in which a high number of simulation runs are required.

4.2.1.1.1 BMW 320 dA

Once the simulation tool has been presented, the specific vehicle model can be described. The simulation model will be a BMW 320 dA. Regarding the model configuration, the definition of a vehicle model in VI-CarRealTime is done using the system files, having a *".xml"* format and stored in vehicle database under *"system.tbl"* table.

The system file contains the following subsystems:

- Front and rear suspension
- Steering
- Body
- Powertrain
- Front and rear wheels and tires
- Brakes
- Auxiliary subsystems

The correlated vehicle model has been provided by the Vehicle Dynamics group of Idiada. The powertrain subsystem has been modified to incorporate an engine per wheel. Additionally, the torque map data has been modified to accomplish a commercial electric car behaviour. This aspect will be described afterwards in Chapter 5. As for the wheel properties, the tire property file contains the data of the Continental Premium Contact 6 225/50 R17 250 kPa tire.

4.2.1.2 Plant model

In addition, in order to simulate an external plant which acts as a real system, that is, it can be used if there is a simulator absence or even to validate its behaviour comparing it with the simulator. Thus, the chassis equations of motion (Zanon et al., 2014) are written as follows

$$\dot{v}_x = v_y \cdot \dot{\psi} + \frac{1}{m}(F_x^{FL} + F_x^{FR} + F_x^{RL} + F_x^{RR}) \tag{4.10}$$

$$\dot{v}_y = -v_x \cdot \dot{\psi} + \frac{1}{m}(F_y^{FL} + F_y^{FR} + F_y^{RL} + F_y^{RR}) \tag{4.11}$$

$$\begin{aligned}\ddot{\psi} = \ddot{\theta} = \frac{1}{Jz}\Big[&(L_F \cdot (F_y^{FL} + F_y^{FR}) - L_R \cdot (F_y^{RL} + F_y^{RR}) \\ &+ \frac{d_F}{2} \cdot (F_x^{FR} - F_x^{FL}) + \frac{d_R}{2} \cdot (F_x^{RR} - F_x^{RL})\Big]\end{aligned} \tag{4.12}$$

where J_z is the moment of inertia of the chassis along z-axis and m is the car mass.

The vehicle longitudinal and lateral accelerations are derived in a straightforward manner

$$a_x = \dot{v}_x - v_y \cdot \dot{\psi} \tag{4.13}$$

$$a_y = \dot{v}_y + v_x \cdot \dot{\psi} \tag{4.14}$$

4.2.2 Tire forces

In this section, a description of the forces and parameters involved to obtain the tire forces will be shown. The longitudinal force of the tire is generated when the vehicle is travelling in a straight line, during acceleration or braking. Then, the lateral force is shown when the vehicle is changing its direction, during cornering. And the vertical tire force is given by the weight of the vehicle and it also depends on the dynamic behaviour of the suspension system.

4.2.2.1 Slip conditions

Before deepen into the longitudinal force aspects, an analysis of using whether pure or combined slip is needed. This study gets simpler if we use as visual support Figure 4.5. Pure slip is defined to be the situation when either longitudinal or lateral slip occurs in isolation (Pacejka, 2012). The force is reduced when the other slip component is added. This situation is known as combined slip. The decrease in force is explained taking into account that the total horizontal frictional force cannot exceed the maximum value (radius of friction circle or ellipse) which is given by the current friction coefficient and normal load.

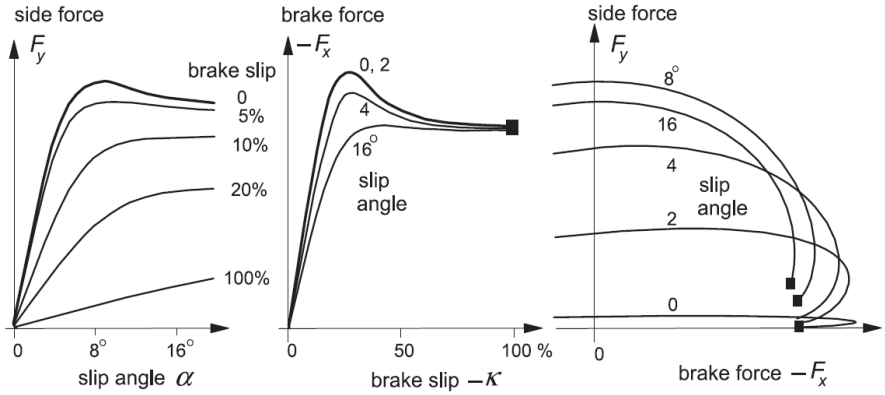


Figure 4.5: Combined side force and brake force characteristics

For simplicity, pure lateral and pure longitudinal slip conditions will be considered (Salehi et al., 2021), (Pacejka, 2012). In order to illustrate the pure slip condition, the brush tire model, a pure longitudinal slip side-view and a pure lateral slip top-view schemes are shown in Figure 4.6. These slip conditions can be present when:

- The longitudinal component of the wheel velocity $v_x = v \cdot cos(\alpha)$ is not equivalent to $v_r = R_e \cdot \omega$, where R_e is the effective radius of the rubber wheel when spinning without an external torque applied to the rolling axis. In this case, longitudinal slip is created.
- The velocity $\vec{v}$, presents an angle α with respect to the tire symmetry plane which is normal to the tire rolling axis. If this situation occurs, lateral slip exists.

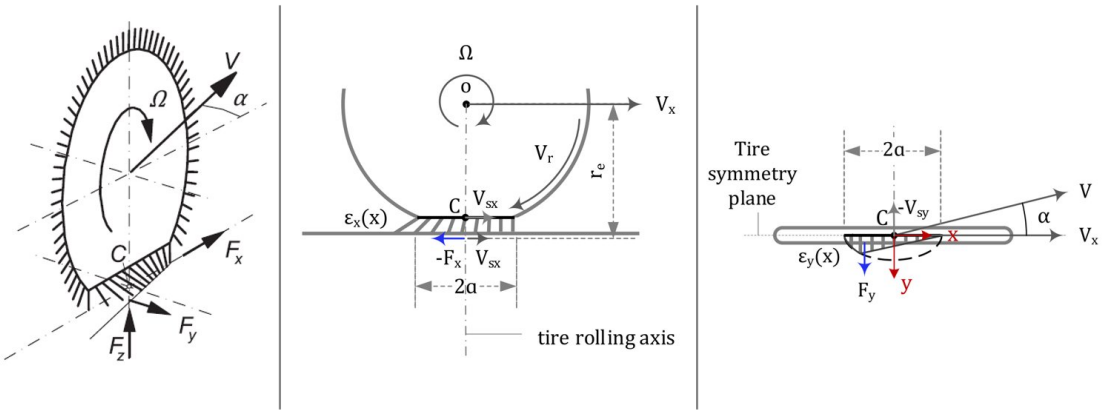


Figure 4.6: Brush tire model, pure longitudinal slip and pure lateral slip

4.2.2.2 Longitudinal dynamics

Regarding the longitudinal forces, the forces are directly computed from the torque applied to the tires through the axle-motor transmission ratio, TR, which translates the

engine torque (directly proportional) and angular velocity (inversely proportional) to the tires. Note that $T_w^{ij} = T_e^{ij} \cdot TR$, where T_w^{ij} and T_e^{ij} correspond to the wheel torque and the engine torque, applied in the front/rear axles, $i = \{F, R\}$, and in the left/right sides, $j = \{L, R\}$.

The longitudinal forces are computed as follows considering the wheel torques and the effective radius of each wheel

$$F_x^{ij} = \frac{T_w^{ij}}{R_e^{ij}} \tag{4.15}$$

A more complex approach could be to use the Pacejka Magic Formula (MF) for the tire longitudinal force either using pure longitudinal slip or combined slip conditions (Pacejka, 2012), or even posing the equilibrium of forces equation along the x-axis (Ghezzi, 2017), as shown in Equation (4.16). However, this approach contains nonlinear terms and implies an increase in computational demand, so the model shown in Equation (4.15) will be implemented instead of it.

Note that the Pacejka's tire model allows to compute the forces and moments acting on a rotating wheel as harmonic functions of the lateral or longitudinal slip ratios. This approach is interesting since the structure of the equations are not provided by any particular physical basis but they are mathematical expressions derived from experimental analysis which fit a wide variety of tire constructions and operating conditions

$$\begin{aligned} \underbrace{m\ddot{x}}_{\text{Inertia force}} &= \underbrace{F_{xf}}_{\text{Front longitudinal drive force}} + \underbrace{F_{xr}}_{\text{Rear longitudinal drive force}} - \underbrace{F_{aero}}_{\text{Aerodynamic drag force}} \\ &- \underbrace{R_{xf}}_{\text{Front rolling resistance force}} - \underbrace{R_{xr}}_{\text{Rear rolling resistance force}} - \underbrace{mgsin\theta}_{\text{Vehicle weight force}} \end{aligned} \tag{4.16}$$

4.2.2.3 Lateral dynamics

The lateral dynamics can be described using both linear and nonlinear models.

The lateral force is called the cornering force when the camber angle (angle between the vertical axis of a wheel and the vertical axis of the vehicle) is equal to zero. At a given tire load, the cornering force increases as the slip angle, α, does it (see Figure 4.7) (Vorotović et al., 2013). In this example, for $\alpha < 8°$ (the specific value will vary from tire to tire), the relationship is linear and the lateral force is given by

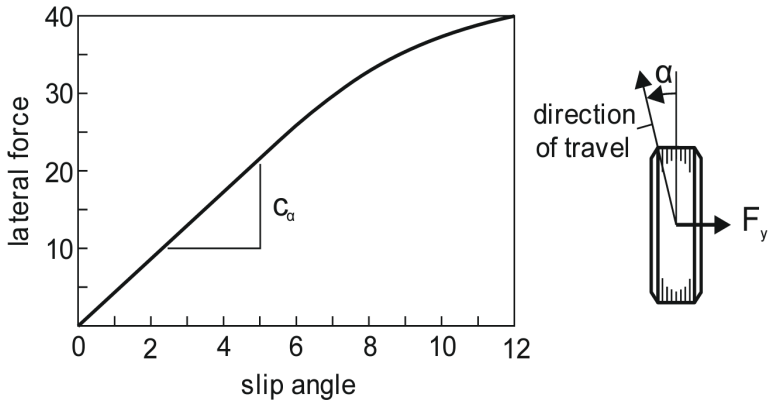


Figure 4.7: Lateral force vs slip angle

$$F_y^{FL} = C_{FL} \cdot \alpha_{FL} \tag{4.17}$$

$$F_y^{FR} = C_{FR} \cdot \alpha_{FR} \tag{4.18}$$

$$F_y^{RL} = C_{RL} \cdot \alpha_{RL} \tag{4.19}$$

$$F_y^{RR} = C_{RR} \cdot \alpha_{RR} \tag{4.20}$$

where C_{ij} and α_{ij} are the cornering stiffness coefficients and slip angles of the front/rear, $i = \{F, R\}$, left/right, $j = \{L, R\}$ tires. Note that the cornering stiffness is defined as the negative of the slope of the curve for F_y versus α at $\alpha = 0$. Moreover, when $\alpha > 0$ yields a negative lateral force on the tire (points to the left) and therefore, a positive cornering stiffness.

The linear model can be very interesting when the slip angle is small. However, when this is not the case a more sophisticated model is required. Another way to describe the lateral tire forces in pure lateral slip is done through the simplified Pacejka Magic Formula (sMF), which consists on a nonlinear relationship between the lateral force, the vertical load and the slip angle with fewer coefficients than the MF (Metzler et al., 2020), (Baselga, 2020). The sMF model enables to optimize the lateral force value from a wide range of vertical load and slip angle curves. The lateral force is computed as follows

$$F_y^{ij} = \mathcal{D}_{ij} \cdot sin\Big(\mathcal{C}_{ij} \cdot tan^{-1}(\mathcal{B}_{ij} \cdot \alpha_{ij})\Big) \tag{4.21}$$

where $\mathcal{D}_{ij}$ represents the peak value of the curve, $\mathcal{C}_{ij}$ is the shape factor and $\mathcal{B}_{ij}$ is the stiffness factor.

The computation of the tire slip angles, α_{ij}, assumes small angles and considers the

track width in each axle

$$\alpha_{FL} = tan^{-1}\left(\frac{v_y + \dot{\psi} \cdot L_F}{v_x - \dot{\psi} \cdot \frac{d_F}{2}}\right) - \delta \tag{4.22}$$

$$\alpha_{FR} = tan^{-1}\left(\frac{v_y + \dot{\psi} \cdot L_F}{v_x + \dot{\psi} \cdot \frac{d_F}{2}}\right) - \delta \tag{4.23}$$

$$\alpha_{RL} = tan^{-1}\left(\frac{v_y - \dot{\psi} \cdot L_R}{v_x - \dot{\psi} \cdot \frac{d_R}{2}}\right) \tag{4.24}$$

$$\alpha_{RR} = tan^{-1}\left(\frac{v_y - \dot{\psi} \cdot L_R}{v_x + \dot{\psi} \cdot \frac{d_R}{2}}\right) \tag{4.25}$$

4.2.2.4 Vertical dynamics

The vertical tire forces are modeled considering the contribution of several terms. The model incorporates the static vertical load in each tire and the load transfer associated with both longitudinal and lateral acceleration in steady-state cornering, including the tangential acceleration and the sideslip rate (Metzler et al., 2020). The vertical forces equations are based on the coefficients C_x, C_{y_R}, and C_{y_F} and several parameters which can be found as follows

$$F_z^{FL} = F_{z,static}^{FL} - C_x \cdot \dot{a}_x - C_{y_F} \cdot v \cdot (\dot{\psi} + \dot{\beta}) \tag{4.26}$$

$$F_z^{FR} = F_{z,static}^{FR} - C_x \cdot \dot{a}_x + C_{y_F} \cdot v \cdot (\dot{\psi} + \dot{\beta}) \tag{4.27}$$

$$F_z^{RL} = F_{z,static}^{RL} + C_x \cdot \dot{a}_x - C_{y_R} \cdot v \cdot (\dot{\psi} + \dot{\beta}) \tag{4.28}$$

$$F_z^{RR} = F_{z,static}^{RR} + C_x \cdot \dot{a}_x + C_{y_R} \cdot v \cdot (\dot{\psi} + \dot{\beta}) \tag{4.29}$$

$$F_{z,static}^{FL} = F_{z,static}^{FR} = \frac{1}{2}\frac{L_R}{L_F + L_R} mg \tag{4.30}$$

$$F_{z,static}^{RL} = F_{z,static}^{RR} = \frac{1}{2}\frac{L_F}{L_F + L_R} mg \tag{4.31}$$

$$C_x = \frac{1}{2}\frac{h}{L_F + L_R} m \tag{4.32}$$

$$C_{y_F} = \left(\frac{L_R}{L_F + L_R}\frac{h_F}{d} + \frac{K_F}{K_F + K_R}\frac{h'}{d}\right) m \tag{4.33}$$

$$C_{y_R} = \left(\frac{L_F}{L_F + L_R}\frac{h_R}{d} + \frac{K_R}{K_F + K_R}\frac{h'}{d}\right) m \tag{4.34}$$

where h is the vehicle center of gravity height; h_F and h_R are the front and rear roll center heights, respectively; h' is the distance between the roll axis and the vehicle center of

gravity; $d = \frac{d_F+d_R}{2}$ is the mean value of the front and rear wheel tracks; and K_F and K_R are the roll stiffness values of the front and rear suspensions, respectively.

4.2.3 Vehicle stability and handling

The vehicle characteristics depend on the relationship between the front tires and their slip angles, α_F and α_R. Additionally, the relationship for the front steering angle δ_F, the turning radius $\mathcal{R}$ around the center of rotation, the wheelbase L, and slip angles is described as follows (Andersson, 2008)

$$\delta_F = \frac{L}{\mathcal{R}} + \alpha_F - \alpha_R \tag{4.35}$$

Note that each tire slip has been previously obtained in Equations (4.22), (4.23), (4.24) and (4.25). The previous relationships give a sense that the steering angle is not only affected by the turning radius but also by the slip angles.

In order to obtain the turning radius, it is necessary to find the center of rotation O of the double-track model (see Figure 4.8), assuming that the steering angle of the front axle δ_F, is the mean of the inner and outer wheel steering angles δ_i, δ_o

$$\delta_F = \frac{\delta_i + \delta_o}{2} \tag{4.36}$$

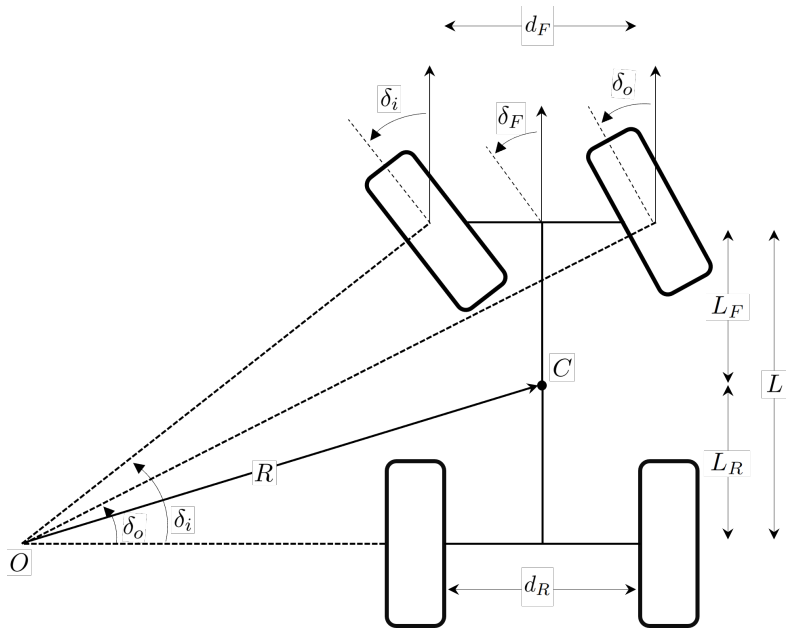


Figure 4.8: Double-track vehicle model for measuring the turning radius

The turning radius is computed below

$$\mathcal{R} = \sqrt{L_R^2 + L^2 cot^2\delta} \tag{4.37}$$

$$cot\delta = \frac{cot\delta_i + cot\delta_o}{2} \tag{4.38}$$

The slip angles difference can be substituted by a term that includes a gain, K_{us}, that is called the understeer coefficient or understeer gradient

$$\delta_F = \frac{L}{\mathcal{R}} + K_{us}\frac{V_x^2}{g\mathcal{R}} = \frac{L}{\mathcal{R}} + K_{us}\frac{a_y}{g} \tag{4.39}$$

The previous equation is crucial for the steady state vehicle handling behaviour. K_{us} is function of the tire cornering stiffness and the weight distribution on the front and rear axles

$$K_{us} = \frac{m_F}{c_{\alpha F}} - \frac{m_R}{c_{\alpha R}} \tag{4.40}$$

Moreover, K_{us} can be measured by means of an understeer gradient test. This step is fully described in Chapter 5. The understeer coefficient depends on the relationship between the front and rear slip angles. The steady state handling characteristics can be divided into three groups according to the value shown by K_{us}:

$$\begin{cases} K_{us} = 0, & \text{Neutral steering} \\ K_{us} < 0, & \text{Oversteering} \\ K_{us} > 0, & \text{Understeering} \end{cases} \tag{4.41}$$

The tendency for a vehicle is to have an understeer behaviour. This is the case when a vehicle turns less than the required steering angle which implies that front tires struggle for adherence. This lost of adherence while cornering, forces the vehicle to go wide on a turn, i.e. moving away from the apex. On the other hand, an oversteer behaviour occurs when the rear tires reach the limit of adhesion in a corner before the front while cornering. Thus, the vehicle turns more than the angle requested by the driver through the steering wheel. In Figure 4.9, these two behaviours can be clearly seen.

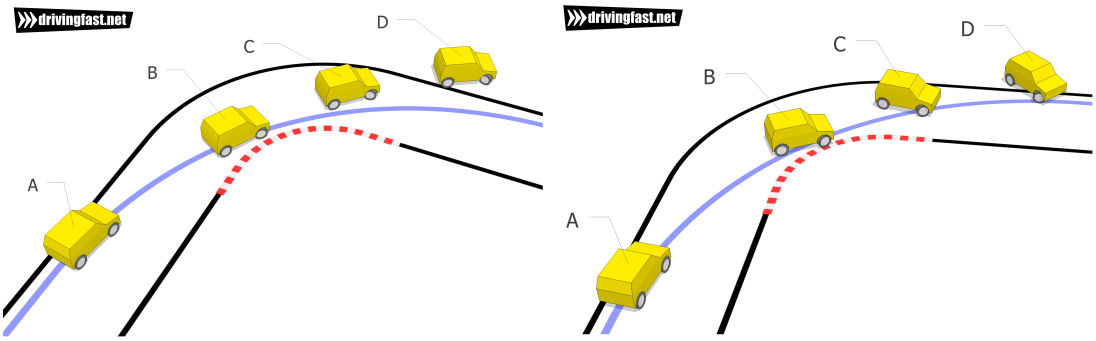


Figure 4.9: Understeering (left) and oversteering (right) behaviours

The characteristic velocity, V_{ch}, is used to describe the phenomenon of a vehicle which understeers, and it is defined as the speed required to perform a curve to double the Ackermann steer angle when the velocity is zero (the front and rear slips α_F and α_R are equal). This means that the vehicle is neutral steered, $\delta_0 = \frac{2L}{R}$.

The characteristic velocity can be determined considering the front and rear wheel cornering stiffness $(c_{\alpha F}, c_{\alpha R})$; and wheelbase distance, L; and the front and rear semi-wheelbase distances (L_F, L_R), respectively (Börner and Isermann, 2005)

$$V_{ch}^2(t) = \frac{c_{\alpha F}(t)c_{\alpha R}(t)L^2}{m\big(c_{\alpha R}(t)L_R - c_{\alpha F}(t)L_F\big)} \tag{4.42}$$

In order to check the stability of a vehicle, from the characteristic velocity equation a stability condition can be extracted according to the Hurwitz stability criterion $(sI - A = 0)$. This leads to the following stability conditions (see Table 4.1).

Stability condition	**Driving situation**	**Stability condition**	**Driving situation**
$1 + \frac{V_x^2}{V_{ch}^2} > 0$	Stable	$V_{ch}^2 > 0$	Understeering
$1 + \frac{V_x^2}{V_{ch}^2} = 0$	Indifferent	$V_{ch}^2 = 0$	Indifferent
$1 + \frac{V_x^2}{V_{ch}^2} < 0$	Unstable	$V_{ch}^2 < 0$	Oversteering
-	-	$V_{ch}^2 \to \infty$	Neutral steering

Table 4.1: Stability and driving conditions

The online computation seen in Equation (4.42) lies in increasing the computational demand. In the literature, the characteristic velocity is mostly considered as a constant parameter (see Equation (4.43)) (Andersson, 2008), (Milliken and Milliken, 1995). The

characteristic speed can be computed once and used as a constant value

$$V_{ch} = \sqrt{\frac{gL}{K_{us}}} \tag{4.43}$$

In Figure 4.10 extracted from (Isermann, 2021), the range of characteristic velocities for cars is shown, ranging from 68 to 112 km/h. The lower bound correspond to a limousine and upper bound to a sports car. Note that the smaller the characteristic velocity, the higher the tendency to understeer.

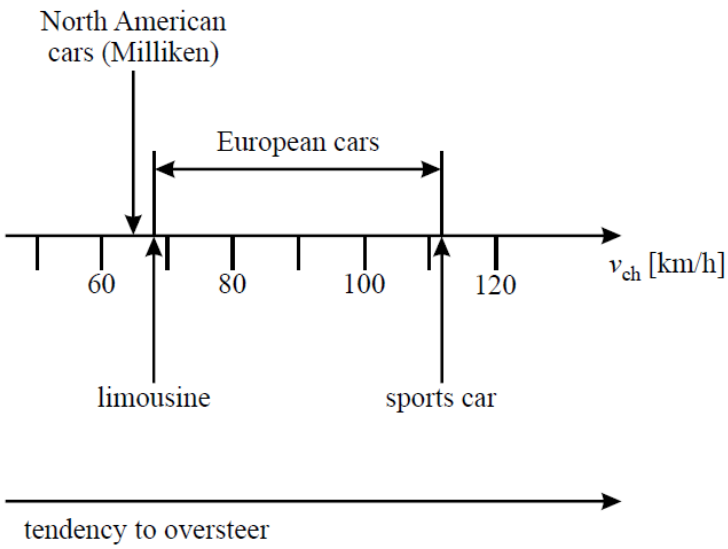


Figure 4.10: Range of characteristic velocities

4.3 Reference generation

In order generate the references to be followed by the vehicle model, a set of considerations must be taken into account.

- Driver inputs: the driver steering and the throttle will be given by the driver, so the generation of references for these signals will not be necessary. The aforementioned inputs will be provided by the driver according to the corresponding maneuver. This step will be described afterwards in Chapter 7.
- Yaw rate reference: the desired yaw rate allows a better lateral dynamics control than if it were not considered.
- Torque request: torque presented by an electric vehicle measured in longitudinal tests at Idiada facilities with a dual motor configuration and which represents the

distribution reference between front and rear axles depending on the speed of the car and the throttle of the driver in situations where lateral distribution is not necessary.

4.3.1 Yaw rate reference

Yaw rate tracking can be formulated mainly following two approaches. The first one considers the steady-state value of the reference yaw rate as a function of the steering angle and the understeering coefficient (Mazzilli et al., 2021). On the other hand, the other approach takes into account the steering angle as well, and the characteristic velocity. In addition, this second approach, incorporates a yaw rate saturation inequality (Chen et al., 2016).

The first approach presents the yaw rate reference, $\dot{\psi}_{ref}$ formulation in Equation (4.44). For a given speed, it brings a linear dependency of the reference yaw rate on steering angle. This formulation is appropriate for systems which intervenes occasionally, i.e considerable yaw rate errors. However, it could be non-ideal for continuously active systems (Lu et al., 2015), since it is desirable for drivers to feel a progressive transition from the linear region to the terminal understeer condition, and therefore, be aware that the driver is approaching the cornering limit.

$$\dot{\psi}_{ref} = \frac{V_x}{L + K_{us} V_x^2} \delta_F \tag{4.44}$$

The formulation of the second approach can be found in Equation (4.45). The reference yaw rate is proportional to the steering wheel angle within a certain range of vehicle velocities when the sideslip angle at CoG is not big. Moreover, the term that is dividing is related to vehicle stability that acts as a inverse gain which reduces the reference yaw rate when the vehicle velocity reaches high velocities, i.e 120 km/h, in order to avoid a high steering wheel sensitivity at high velocities. The desired yaw rate is limited by the maximum force the road can provide through the road friction coefficient, μ_r (see Equation (4.46)).

$$\dot{\psi}_{ref} = \frac{V_x}{L\left(\frac{V_x}{V_{ch}}\right)^2} \delta_F \tag{4.45}$$

$$|\dot{\psi}_{ref}| = \left|\frac{\mu_r g}{V_x}\right| \tag{4.46}$$

In Figure 4.11, there exists a comparison of both approaches for a ramp steer maneuver

in which the steering angle increases at 80 and 120 km/h. The yaw rate reference is quite similar between the approaches and velocities until reaching 0.45 rad/s but as the steering angle increases the saturation approach yields a lower (saturated) value resulting in a lower steering wheel sensitivity while the first approach still increases the yaw rate reference. Hence, the second method will be the one used for the simulations.

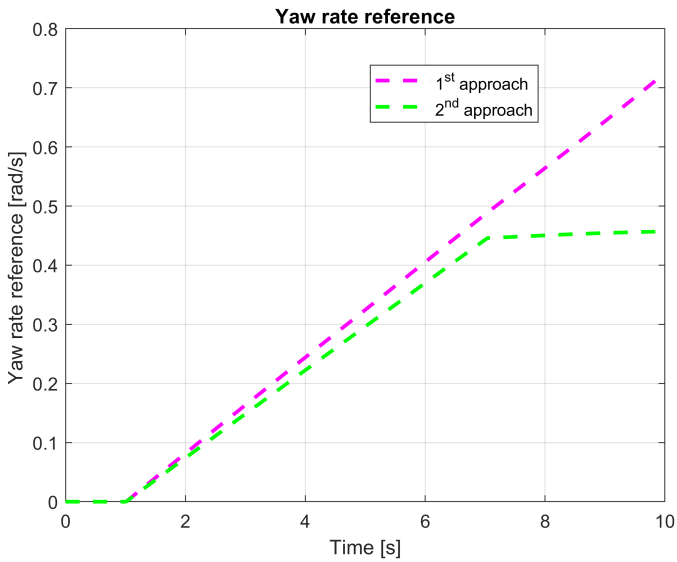


Figure 4.11: Comparison of yaw rate reference generation approaches

4.3.2 Torque request

The torque request generator is essential since the Torque Vectoring strategy considers the distribution reference. In fact, this distribution is taken into account in the cost function of the optimization problem and is modified by a certain gradient in order to meet the rest of the objectives of the cost function, such as the reference yaw rate. Moreover, the torque gradient with respect to the reference is not given by axle between right and left, but simultaneously to the four wheels so that the total torque request remains the same as that initially computed. The cost function of the NMPC problem that will be described in Chapter 6.

The torques are obtained through a 2D LookUp table for the front and rear engine torques whose inputs are the motor angular velocity and the percentage of throttle applied. First, we need to convert the wheel angular velocity to the engine and then, the engine torque to the wheels. The Simulink scheme is shown in Figure 4.12

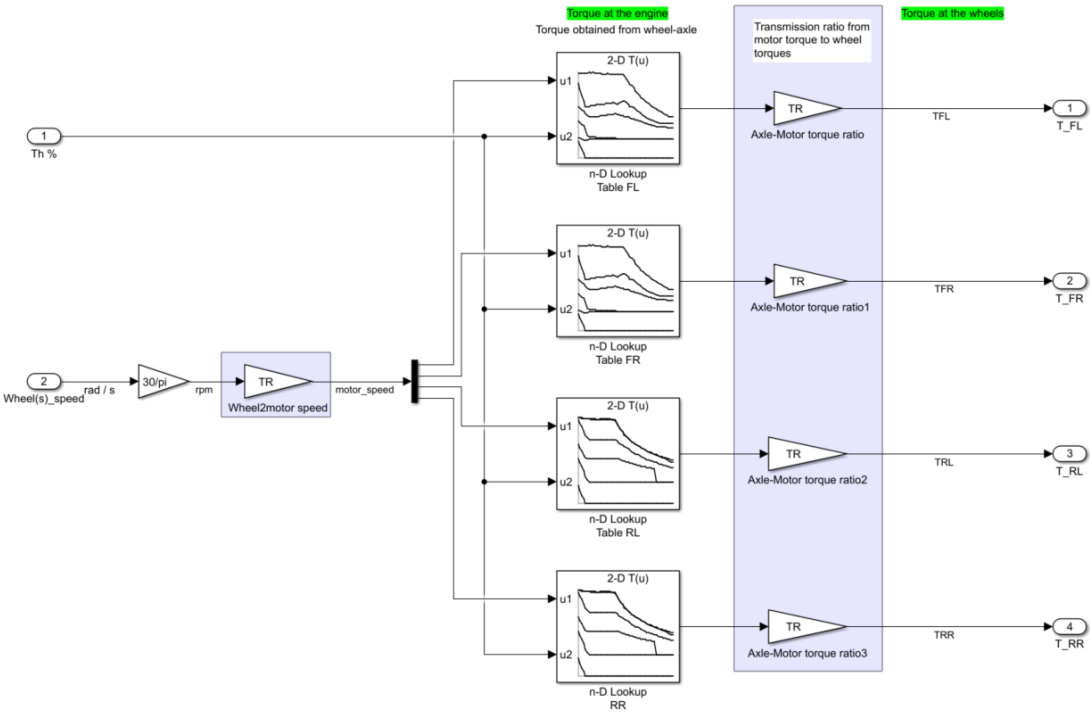


Figure 4.12: Torque request generation

In order to be able to predict correctly lateral forces a passive (without controllers) correlation has been done between VI-CarRealTime values and Simplified Magic Formula used in the NMPC plant. This step will be described in Chapter 5. The LookUp table uses the current velocity and the throttle input to generate a total torque request.

Chapter 5

Parameter Estimation

In this chapter, the parameters involved in the vehicle dynamics formulations and in the generation of references will be identified. The parameters can be obtained through two different approaches:

1. Physical measurements: both data catalogues and physical measurement are included in the real measurable parameter group.
2. Parameter estimation: include the parameters that cannot be directly obtained from sensor data but applying intermediate computations or parameter estimation techniques to measured data.

In a real case scenario, i.e. using a car such as the BMW 320 dA, the dynamic equations of the vehicle are not available, so parameter estimation is essential. Therefore, the unknown parameters must be adjusted with real operating data.

On the one hand, in Section 5.1, the parameters are obtained from direct measurements. On the other hand, Section 5.2 is devoted to the description of the parameter identification steps using.

5.1 Physical Measurements

The measured parameters have been obtained using the ZePerfs catalogue (ZePerfs, 2021) and the sensing of parameters during real maneuvers at Applus IDIADA proving ground (test tracks). The measured parameters can be seen in Table 5.1

Symbol	Description	Value	Unit
m	vehicle mass (kerb weight)	1619.4	kg
J_z	vehicle moment of inertia along z-axis	2807	$kg\,m^2$
L_F	distance from CoG to front axle	1.385	m
L_R	distance from CoG to rear axle	1.466	m
L	vehicle wheelbase	2.851	m
d_F	front wheel track	1.570	m
d_R	rear wheel track	1.585	m
R_u	unloaded radius	0.3298	kg
μ_r	road friction coefficient (dry)	0.9	–
g	gravity constant	9.81	$\frac{m}{s^2}$
V_{ch}	characteristic velocity	110	$\frac{km}{h}$
K_{us}	understeer coefficient	12.2215	$\frac{deg}{\frac{m}{s^2}}$

Table 5.1: Vehicle parameters

5.2 Parameter Estimation

5.2.1 Least-squares fitting

The goal of the least-squares (LS) method consists of adjusting the parameters of a model function to optimally fit a data set minimizing the sum of squares of the differences in the ordinates between the points generated by the chosen function and the corresponding values in the data. So, the aim lies in finding the vector of parameters given a set of experimental input and output data so that it minimizes the cost function (MathWorks, 2022c). The following optimization problem is formulated

$$\min_{\theta} ||J_N||^2 = \frac{1}{N}\sum_{k=0}^{N}(\varphi^T\theta - y_k)^2 \tag{5.1}$$

where $y_m = \varphi^T\theta$ is the estimation model, φ is the vector of inputs, θ are the parameters to be identified and y_k is the set of measured outputs. Moreover, Equation (5.1) can be solved analytically if the inverse of $(\varphi^T\theta - y_k)$ exists due to both the parameterization linearity and that the parameter vector implies that the cost function becomes quadratic. The least-square estimate is given by

$$\hat{\theta}_N = \min \frac{1}{N^2}\left[\sum_{k=0}^{N}\varphi\,\varphi^T\right]^{-1}\left[\sum_{k=0}^{N}\varphi\,y_i\right] \tag{5.2}$$

Additionally, both linear and nonlinear least-squares problems can be addressed depending on the characteristics of the model selected to fit the observations.

5.2.2 Parameter identification

Once the fitting technique is formulated, the parameter identification step can be described with the particularities of the different parameters.

5.2.2.1 Understeer coefficient

Regarding the parameters that cannot be directly obtained from input data, an experiment must be carried out to measure the understeer coefficient, K_{us}.

A ramp steer maneuver is performed with a constant longitudinal speed of 30 km/h and increasing the steering wheel angle, SWA, from 0 to 60 degrees. Note that this gradient test could be also performed with a SWA ranging from -30 to 30 degrees but as the vehicle is identical from the point of view of the longitudinal axis this SWA is not necessary. The understeering gradient can be computed taking into account the partial derivative of the steering angle with respect to the lateral and evaluated at zero lateral acceleration (Tse, 2018)

$$K_{us} = \left.\frac{\partial \delta}{\partial a_y}\right|_{a_y=0} - \frac{L}{V_x^2} \tag{5.3}$$

It is crucial to have a longitudinal speed that is around 30 km/h and does not differ much from this value during the test. This speed is relatively low compared to a general highway driving speed, precisely with the aim of reducing the likelihood of a loss-of-control.

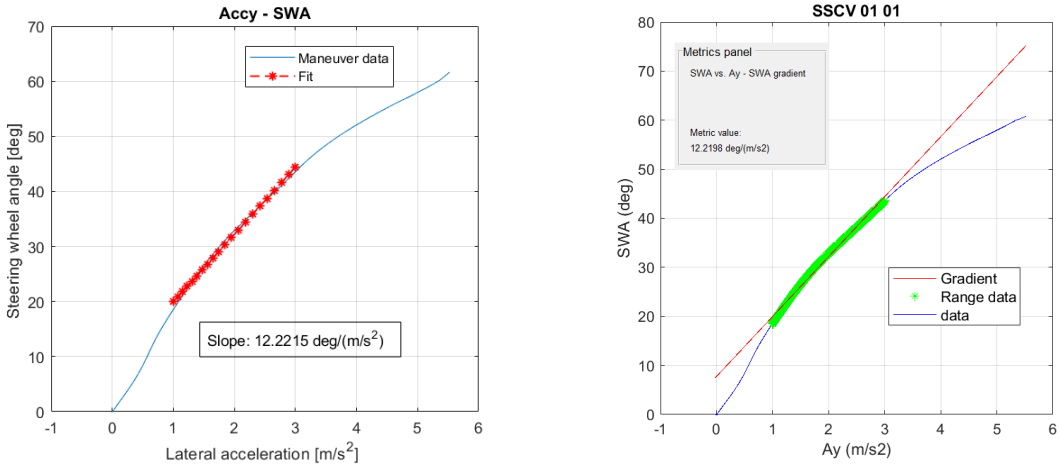
This maneuver will be generated in VI-CarRealTime as well as the NMPC validation maneuvers that will be described in Chapter 8.

The first step lies on identifying the slope of the lateral acceleration (x-axis) and steering angle (y-axis). The computation of the slope at zero lateral acceleration is done by means of a linear least-squares fitting for the lateral acceleration range (1 - 3) $\frac{m}{s^2}$. The slope of the first-order polynomial is

$$\left.\frac{\partial \delta}{\partial a_y}\right|_{a_y=0} = 12.2215 \ \frac{deg}{\frac{m}{s^2}} \tag{5.4}$$

This fitting has been validated using a SWA gradient metric test using a tool provided by IDIADA that uses the real car model leading to 12.2198 $\frac{deg}{\frac{m}{s^2}}$. In Figure 5.1 both

fittings are shown.



(a) Least-squares fitting (b) SWA gradient validation

Figure 5.1: SWA gradient computation

Once the slope has been computed and validated, the understeer gradient is obtained

$$K_{us} = 9.8693 \, \frac{deg}{\frac{m}{s^2}}, \quad K_{us} > 0 \rightarrow \text{Understeer behaviour} \tag{5.5}$$

Moreover, the characteristic velocity, V_{ch}, is obtained performing driving tests in the Proving ground of IDIADA instead of computing the Equation (4.43) which would have been less reliable than doing a test with the real vehicle, $V_{ch} = 110 \; km/h$. Therefore, considering the K_{us} and V_{ch} values, the stability conditions are verified leading to an stable driving situation $(1 + \frac{V_x^2}{V_{ch}^2} > 0)$ where the car understeers $(V_{ch}^2 > 0)$.

5.2.2.2 Pacejka coefficients

In order to be able to predict correctly lateral forces, a correlation has been done between VI-CarRealTime data and Simplified Magic Formula used in Equation (4.21) for the lateral forces. The non-linear least-squares (NLSQ) method will be useful to estimate the set of coefficients $\{d, b, c\}$ from the lateral force which depends on the tire slip angles $F_y^{ij}(\alpha_{ij})$. Note that VI-CarRealTime has a tool called *Tire Testrig* where the user can simulate the tire forces under specified conditions for a range of values that resultant lateral forces depend on.

In this case, the camber angle which is the angle between the vertical axis of a wheel and the vertical axis of the vehicle, and vehicle speed are variables that are considered

invariant on the NMPC model and only the vertical force and lateral sideslip angles are the ones responsible for varying the lateral force. For a range of -15 to 15 degrees slip angles and vertical loads from $50\,N$ to $8\,kN$ (each $883\,N$) the obtained curves can be seen in Figure 5.2

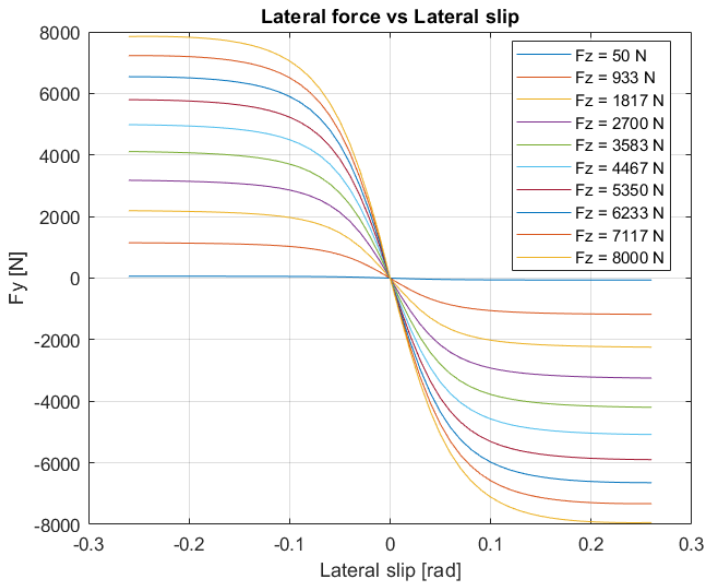


Figure 5.2: Lateral force vs slip angles

The dataset has a size of $10 \times 2 \times 5 \cdot 10^6$ pair samples, so it must be filtered to work properly. The filtering consists on two stages:

1. Select the unique data pairs and compute the mean of the hysteresis values, yielding to a lateral force unique value for a given slip angle.

2. Round up the slip angles data to two decimals so that the dataset is reduced and the resolution of the slip angles is still representative ($0.01\,rad = 0.57$).

After filtering, the dataset is reduced to $10 \times 2 \times 53$ samples. Now, the nonlinear least-squares fitting is used to obtain the matrices of parameters for the slip angles and lateral forces pairs. For this purpose, the fitting has been divided into two slip angles intervals $[-15, 0]$ and $[0, 15]$. Thus, the curve fitting will be more accurate since it is applied for the negative and positive slip angles and then merged for $\alpha = 0$. The fitting for all the cases has been tuned by means of the initial values and tolerance to reduce the error given by the model and measures difference. The fitting can be seen in Figure 5.3 where MF(-) and MF(+) stands for the Magic Formula lateral force values using the coefficients estimated for the negative and positive slip angles, respectively.

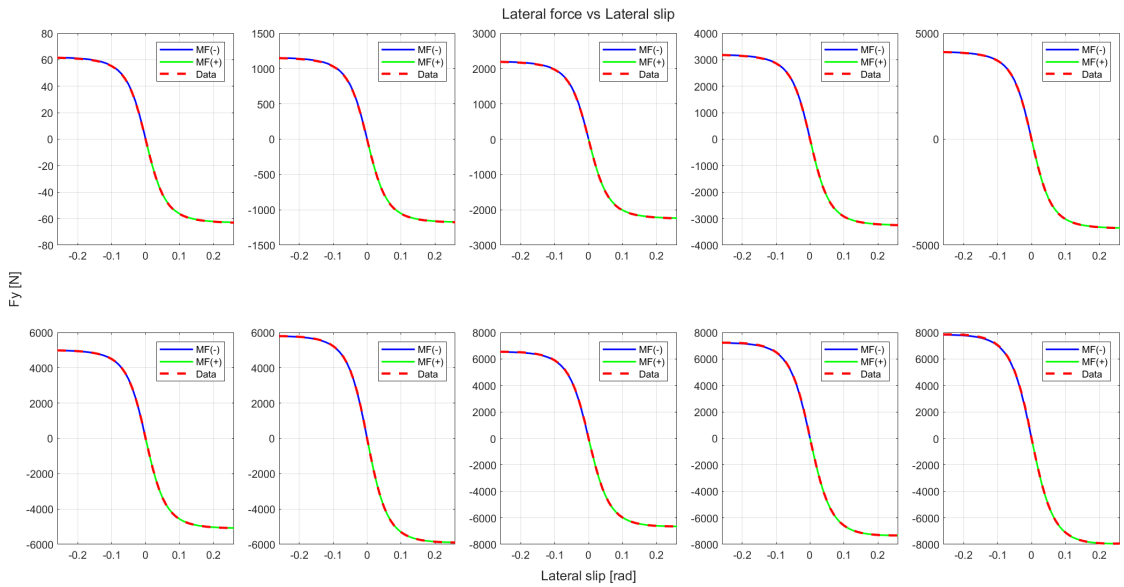


Figure 5.3: Lateral force fitting through Pacejka coefficients

5.2.2.3 Tires radius

Another important aspect to be analyzed is the radius of each tire, the tires radius vary over time due to several factors such as the car load, the surface of the contact patch, the camber angle, etc. However, the unloaded radius, R_u can be compared with the evolution of the dynamic effective radius, R_e in order to see if the difference is negligible letting to use a constant rather that computing each time step the radius of each tire. The vehicle plant provided by VI-CarRealTime allows to access the car output data during a test. Thus, a comparison between the effective radius computed each time step and the unloaded radius is shown in Figure 5.4.

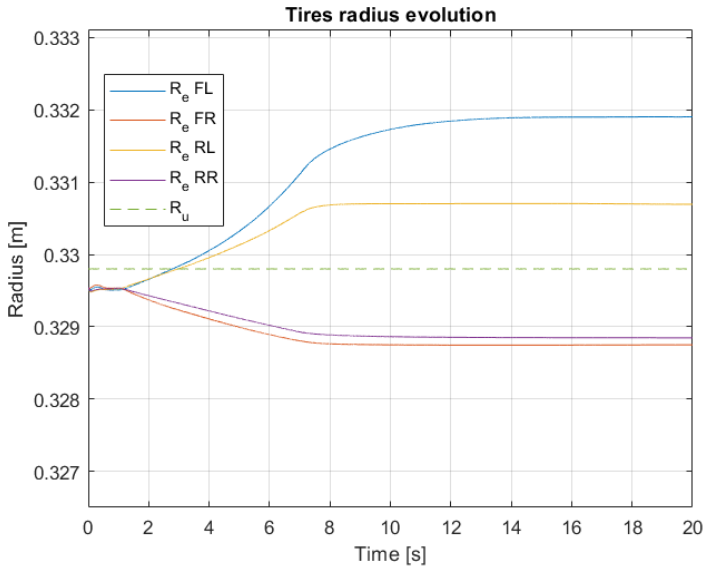


Figure 5.4: Tires radius evolution

It can be seen that the dotted line, R_u is close to the effective radius in each wheel. The maximum error found is 2.1 mm with respect to the front left tire which is assumed negligible. Therefore, the unloaded radius will be enough to be incorporated to the vehicle model. Likewise, the computation load decreases by not computing the effective radius each time.

5.2.2.4 Torque pedal map

The car performance depends on the torque available, it is provided by each electric engine located at the left and right side of both front and rear axles. In order to build a torque pedal map, firstly, the engine curves are necessary to know the relationship between the engine angular speed (rpm) and the torque (Nm).

For this purpose, the Tesla Model S whose torque is around 640 Nm has been to used to extract the car torque map. This vehicle has 2 electric engines located at the front and rear axle (dual motor configuration). Thus, in order to build a four-in-wheel motor drive scheme, each engine will be split giving rise to the scheme shown in Figure 5.5 where each wheel is directly related with only a single engine.

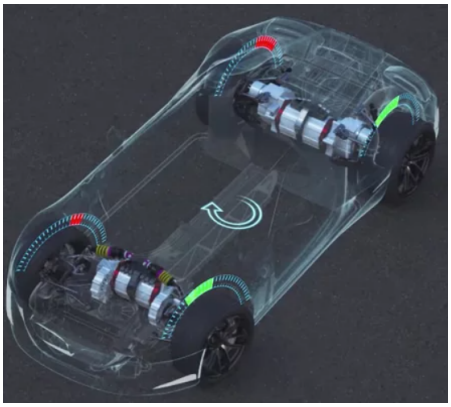


Figure 5.5: Four in-wheel motor drive scheme

It is necessary to redistribute the torque of each axle to every single engine. Firstly, the distribution is set to 35:65 regarding front and rear axles. Then, the torque map is obtained from a matching different signals (throttle, torque, power, gear, etc) from a dyno (bench) test. In Figure 5.6, both front and rear axle torques are shown.

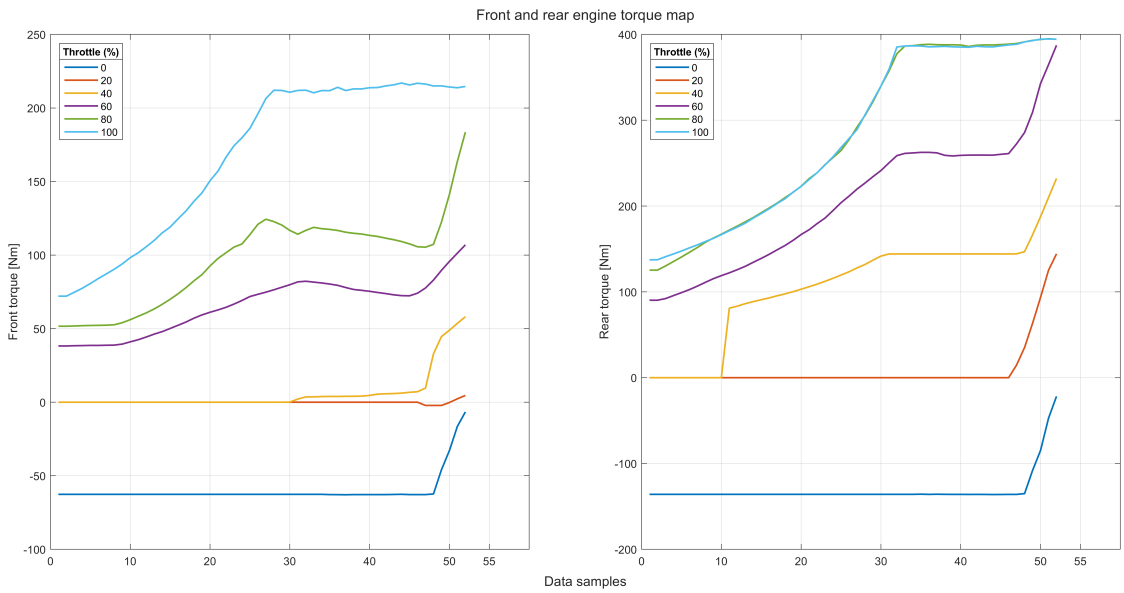


Figure 5.6: Front and rear axle torque map

Then, setting a 50:50 torque distribution takes place, for the each engine with respect with the initial dual engine configuration at front and rear axles. The front left and right, and the rear left and right torque maps can be seen in order of appearance in Figure 5.7.

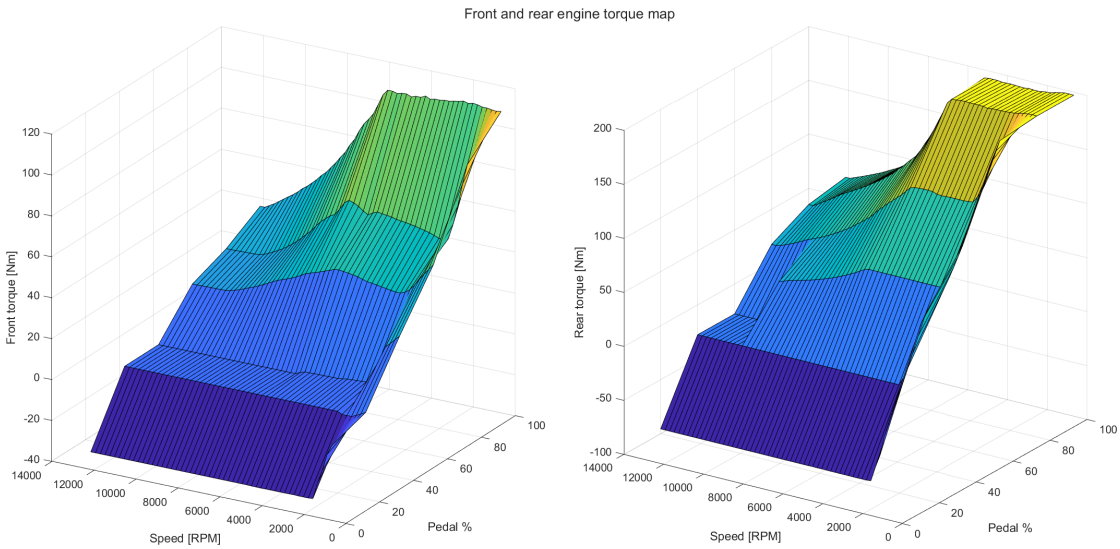


Figure 5.7: Front (left and right) and rear (left and right) torque pedal map

To validate if the identified parameters are both feasible and reliable, several maneuvers are performed. This step will be described in detail afterwards in Chapter 7.

Chapter 6

Torque Vectoring design

The section addresses the real-time control of the electric vehicle applying an NMPC controller. The vehicle control inputs vary every optimization step over the prediction horizon based on the vehicle model and constraints. The developed algorithm is validated in two stages: first, in a simulated scenario and afterwards, in a real-world scenario testing different driving maneuvers in both stages. Besides the previous aspects, computation time in each optimization window is monitored so that it does not exceed the sampling time and the real-time specification is fulfilled.

In the next Section 6.1, a nonlinear double-track vehicle is developed. In Section 6.3, the objective function with constraints are described as well as their significance. Then, in Section 6.4 the solvers particular formulations are developed. Moreover, a deeper analysis of the maneuvers and scenarios is carried out in Chapter 8.

6.1 Internal plant model

The prediction model is a double-track vehicle model, with yaw rate, $\dot{\psi}$, and sideslip angle, β, as states and the wheel torques as inputs, $[T_{FL}, T_{FR}, T_{RL}, T_{RR}]$. The nonlinear model structure is presented below

$$\dot{x} = f(x, u) \tag{6.1}$$

where

$$x = \begin{bmatrix} \beta \\ \dot{\psi} \end{bmatrix}, \qquad u = \begin{bmatrix} T_{FL} \\ T_{FR} \\ T_{RL} \\ T_{RR} \end{bmatrix}$$

The continuous second order nonlinear model is shown as follows

$$\begin{aligned}\dot{\beta} = \frac{1}{mv_x}\Big[&(T_{FL}+T_{FR})\cdot\frac{1}{R_u}\cdot sin(\delta-\beta)+(F_y^{FL}+F_y^{FR})\cdot cos(\delta-\beta)\\ &-(T_{RL}+T_{RR})\cdot\frac{1}{R_u}\cdot sin(\beta)+(F_y^{RR}+F_y^{RR})\cdot cos(\beta)\Big]-\dot{\psi}\end{aligned} \tag{6.2}$$

$$\begin{aligned}\ddot{\psi} = \frac{1}{J_z}\Big[&(T_{FL}+T_{FR})\cdot\frac{1}{R_u}\cdot sin(\delta)\cdot L_F+(F_y^{FL}+F_y^{FR})\cdot cos(\delta)\cdot L_F\\ &-(T_{RL}+T_{RR})\cdot\frac{1}{R_u}\cdot L_R-(F_y^{FR}-F_y^{FR})\cdot cos(\delta)\cdot\frac{d_F}{2}\\ &+(T_{FL}-T_{FR})\cdot\frac{1}{R_u}\cdot sin(\delta)\cdot\frac{d_F}{2}-(F_y^{RL}-F_y^{RR})\cdot\frac{d_R}{2}\Big]\end{aligned} \tag{6.3}$$

6.2 NMPC algorithm

The NMPC pseudocode can be seen in Algorithm 1, which provides a high level overview of the real-time deployment. The system setup is initialized by the measures of the vehicle states, inputs (null at time $t = 0$), driver demands (driver steering, δ, and brake signals, B) and parameters.

Algorithm 1 Torque Vectoring NMPC Algorithm

Input: Current vehicle states (x_k), past control inputs (u_{k-1}), driver steering (δ_k) and brakes (B_k)
Output: Vehicle states predicted and wheels torque over the prediction horizon (z_{k+1}), $\forall k = 1, \ldots, H_p$
$x_0 \leftarrow$ *vehicle model states initialization*
$p_0 \leftarrow$ *input parameter initialization*
MPC setup
$Q_{e_x}, Q_{r_u}, Q_{e_u}, Q_{e_\xi}, Q_{e_{\tilde{u}}}$ Weights setting
while *Vehicle is moving* **do**
  $\hat{x} \leftarrow$ *update vehicle model states*
  $\hat{p} \leftarrow$ *update parameters*
  $[\hat{\underline{z}}, \hat{\overline{z}}] \leftarrow$ *update optimization bounds*
  $[\hat{\underline{h}}, \hat{\overline{h}}] \leftarrow$ *update inequalities bounds*
  $\vec{x}, \vec{u} \leftarrow NMPCsolve(\hat{x}, u_{-1}, \hat{p})$
  $u_{-1} \leftarrow \vec{u}[0]$
  Apply control action $\vec{u}[0]$
end

There are also two key aspects that must be considered. Firstly, it is necessary to set a initial guess to start the solver at time ($t = 0$) and secondly, it is crucial to set initial conditions for each problem instance.

The objective terms, the equality and inequality constraints (f_k, c_k, h_k) are generated over the prediction horizon by means of the current parameters, p_k, vehicle states, previous inputs and selection matrix, E_k. The *NMPCsolve()* function optimizes the optimal-control sequence to be applied to the system over the horizon to follow the reference trajectory that for instance can be provided by the driver demands once a maneuver has been designed. At the same time, the *NMPCsolve()* function minimizes the objective function and meets the constraints.

6.3 NMPC formulation

The NMPC-based vehicle Torque Vectoring control is focused on achieving real-time performance to enable implementable solutions in the real world.

The NMPC strategy works with a frequency of 100 Hz, that is, the sampling time is, $T_s = 10\,ms$ which means that plant $(T_s = 1\,ms)$ updates the system variables 10 times faster than the controller. If a more exhaustive time constraint is desired it is necessary to reduce the sampling time of the controller but the real-time specification cannot be exceeded.

On the other hand, the prediction horizon, H_p, is the length of time into the future or the number of future time steps, which in this case is set to 5 steps that correspond to $50\,ms$. Regarding the selection of $H_p = 5$, there is a trade-off between the controller being reactive and its control effort not being too great and at the same time not to overpass the real-time constraint.

6.3.1 Cost function

The objective function J_k for the NMPC problem $(N = H_p)$ is formulated as a quadratic optimization problem that is solved at each time k to determine the control actions considering that the values of x_k and u_{k-1} are known

$$\min_{\Delta u_k, u_k, x_k} J_k = \overbrace{\sum_{k=0}^{N-1} \left(x_{\text{ref}} - x_k\right)^\top Q_{e_x} \left(x_{\text{ref}} - x_k\right)}^{J_{Q_x}} + \overbrace{\sum_{k=0}^{N-1} \Delta u_k^\top Q_{r_u} \Delta u_k}^{J_{\Delta u}} +$$
$$+ \overbrace{\sum_{k=0}^{N-1} \left(u_{\text{ref}} - u_k\right)^\top Q_{e_u} \left(u_k - u_{\text{ref}}\right)}^{J_u} + \overbrace{\sum_{k=0}^{N-1} \left(e_{T_{req}}\right)^\top Q_{e_\xi} \left(e_{T_{req}}\right) + \lambda \xi_T}^{J_{\xi_T}} +$$
$$+ \overbrace{\left(u_{(k=0)} - u_{-1}\right)^\top Q_{e_{\tilde{u}}} \left(u_{(k=0)} - u_{-1}\right)}^{J_{\tilde{u}}} \Bigg|_{k=0} \tag{6.4a}$$

$$\text{s.t.} \quad x_{k+1} = f(x_k, u_k) \tag{6.4b}$$
$$\Delta u_{\min} \leq \Delta u_k \leq \Delta u_{\max} \tag{6.4c}$$
$$u_{\min} \leq u_k \leq u_{\max} \tag{6.4d}$$
$$x_{\min} \leq x_k \leq x_{\max} \tag{6.4e}$$
$$\Delta \tilde{u}_{\min} \leq \left(u_{(k=0)} - u_{-1}\right) \leq \Delta \tilde{u}_{\max} \tag{6.4f}$$
$$-T_{req} + \xi_T \leq \left(T_{FL} + T_{FR} + T_{RL} + T_{RR}\right) \leq T_{req} - \xi_T \tag{6.4g}$$
$$x_0 = \bar{x} \tag{6.4h}$$
$$u_{-1} = \bar{u} \tag{6.4i}$$
$$\xi_T \geq 0, \; \lambda \geq 0 \tag{6.4j}$$
$$\forall k = 0, \ldots, N, \tag{6.4k}$$

where $e_{T_{req}} = T_{req} - \left(T_{FL} + T_{FR} + T_{RL} + T_{RR}\right)$

$$\Delta \tilde{u}_{\min} = \begin{bmatrix} \Delta \tilde{u}_{\min}^{FL} \\ \Delta \tilde{u}_{\min}^{FR} \\ \Delta \tilde{u}_{\min}^{RL} \\ \Delta \tilde{u}_{\min}^{RR} \end{bmatrix} = \begin{bmatrix} 0.35 \cdot \Delta u_{\min} \\ 0.35 \cdot \Delta u_{\min} \\ 0.65 \cdot \Delta u_{\min} \\ 0.65 \cdot \Delta u_{\min} \end{bmatrix}, \quad \Delta \tilde{u}_{\max} = \begin{bmatrix} \Delta \tilde{u}_{\max}^{FL} \\ \Delta \tilde{u}_{\max}^{FR} \\ \Delta \tilde{u}_{\max}^{RL} \\ \Delta \tilde{u}_{\max}^{RR} \end{bmatrix} = \begin{bmatrix} 0.35 \cdot \Delta u_{\max} \\ 0.35 \cdot \Delta u_{\max} \\ 0.65 \cdot \Delta u_{\max} \\ 0.65 \cdot \Delta u_{\max} \end{bmatrix}$$

$$Q_{r_u} = \begin{bmatrix} 0.35 & 0 & 0 & 0 \\ 0 & 0.35 & 0 & 0 \\ 0 & 0 & 0.65 & 0 \\ 0 & 0 & 0 & 0.65 \end{bmatrix} \cdot Q_{r_u}^*,$$

where Q_{e_x} is the yaw rate error weight; Q_{r_u} is the torque rate weight; Q_{e_u} is the torque error weight; Q_{e_ξ} is the torque request error weight; and $Q_{e_{\tilde{u}}}$ is the input change $(u_{(k=0)} - u_{-1})$ error weight. Then, $\Delta u_k = (u_k - u_{k-1}) \cdot T_s^{-1}$ is the torque rate and $\bar{x}$ is the initial state at the beginning of each horizon. The optimization variable are the elements of the input sequence

$$U = \{u_0,\ u_1,\ \ldots,\ u_{N-1}\} \tag{6.5}$$

The developed cost function for the NMPC problem consists of five cost functions (see Equation (6.4)). The first term consists of the yaw rate error, $J_{Q_x} = J_{Q_{\dot{\psi}}}$, to minimize the error to reach the reference value.

The second term in the cost formulation is related to the torque rate, $J_{\Delta u}$, ensures the change in input to be smooth also called as slew rate.

The third term takes into account the torque error, J_u. This way, the input cost function tries to minimize the effort or energy input to the system.

The fourth term consists of the torque request error, J_{ξ_T}. This term tries to minimize the difference between the sum of all wheel torques and the request torque at each time step. In addition, we add a linear penalty term inside the fourth term to punish large positive ξ_T values using the hyperparameter $\lambda \geq 0$. Here, a penalty term linear in ξ_T is used.

Moreover, there exists an extra term that only applies in the first iteration of each optimization window related to the error of the current input with respect to the optimized one of the previous optimizer call, $J_{\tilde{u}}$, which aims to smooth the input change $(u_{(k=0)} - u_{-1})$ through successive optimizer calls in order not to have input noticeable offsets over time which could cause that there is no continuity of the torques, yielding to difficulties in the optimization.

The formulated optimization problem is a nonlinear problem due to the nonlinear dynamics of the system. The predicted states over the finite prediction horizon are computed from the current states and the predicted control inputs that take into account the previous solution.

The NMPC input is applied following a receding horizon fashion (Charitopoulos and Dua, 2016) that involves repeatedly solving a constrained optimization problem, using predictions of future costs, and constraints over a moving time horizon to choose the control action. At each time step the previous formulation is solved with $x_0 = x(k)$, control input $u_{-1} = u(k-1)$ and an optimal sequence $u = \{u_0, \ldots, u_{N-1}\}$ is obtained.

The first input u_0 provided by the optimal control sequence is applied to the system at time k. Then, at time instant $k = k+1$, the system evolved at new state $x(k+1)$ which is measured or estimated and the whole process is repeated until reaching the prediction horizon length (Chaubey, 2021). Thus, the control law depends on both the current state and the previous input (and the gain, K) as follows

$$u = K\big(x(k),\, u(k-1)\big) \tag{6.6}$$

Moreover, a set of proportionality constants $\{0.35, 0.65\}$ for the front and rear wheels, respectively, has been added to Q_{r_u} and $\{\Delta\tilde{u}_{\min}, \Delta\tilde{u}_{\max}\}$ in order to the include the engine torque difference between front and rear wheels. The engines placed at the front wheels contribute a 35% and the rear wheel engines a 65% of the total torque provided by the vehicle.

6.3.1.1 Constraints

In the NMPC formulation (6.4), there are constraints whose meaning will be described in the following sections:

6.3.1.1.1 Hard Constraints

These constraints are the ones that the solver solution must satisfy. If it is not satisfied at a given iteration, the solution is infeasible and therefore, the manipulated variables which are the torques will be set with the previous step value, that is, $u(k) = u(k-1)$. Thus, if the infeasibility is not resolved in the next steps it can continue indefinitely and that can finally cause a loss of control (MathWorks, 2022b).

The model equalities (see Equation (6.4b)) involve the lateral force and yaw moment balance equations (see Equations (6.2) and (6.3)); the slip angles found in Equations (4.22), (4.23), (4.24) and (4.25); and the lateral forces found in Equation (4.21). In the solvers section (see Section 6.4), some extra terms are added due to the particularities of the solver. These equations represent the vehicle dynamics and must hold under the prediction horizon so that the dynamics of the internal plant of the controller reproduces what is observed in simulation plant for any maneuver.

Equations (6.4c), (6.4d), (6.4e), (6.4f) are also hard constraints that must be fulfilled. Regarding the lower and upper bounds for the torque applied to each wheel (in $[Nm]$), there are physical constraints that limit the engine torques, T_e. The lower bound values are provided by the minimum values allowed for each wheel, $\underline{T_w}$, in the BMW 320 dA

pedal map database

$$\underline{T_w} = \begin{bmatrix} \underline{T_w^{FL}} \\ \underline{T_w^{FR}} \\ \underline{T_w^{RL}} \\ \underline{T_w^{RR}} \end{bmatrix} = \underline{T_e} \cdot TR, \qquad \underline{T_e} = \begin{bmatrix} -32 \\ -32 \\ -70 \\ -70 \end{bmatrix} \tag{6.7}$$

On the other hand, the torques upper bound is limited by the minimum value obtained for three different conditions (Ghezzi, 2017):

$$\begin{aligned} T_{max}^{ij} &= min(\text{Pedal map}, \text{Road adhesion}, \text{Friction ellipse}) \\ &= min(\overline{T_w},\ |T_{max,adh}^{ij}|,\ |T_{max,frict}^{ij}|) \end{aligned} \tag{6.8}$$

- The Pedal map of the engine torque, T_e, limits the maximum torque to be applied to each wheel, $\overline{T_w}$:

$$\overline{T_w} = \begin{bmatrix} \overline{T_w^{FL}} \\ \overline{T_w^{FR}} \\ \overline{T_w^{RL}} \\ \overline{T_w^{RR}} \end{bmatrix} = \overline{T_e} \cdot TR, \qquad \overline{T_e} = \begin{bmatrix} 108 \\ 108 \\ 200 \\ 200 \end{bmatrix} \tag{6.9}$$

- The Road adhesion coefficient involving the vertical load F_z^{ij}, the longitudinal friction coefficient, μ_x^{ij} and the wheel radius, R:

$$|T_{max,adh}^{ij}| = \mu_x^{ij} F_z^{ij} R \tag{6.10}$$

- The last condition comes from the compliance of the Friction ellipse equation on the tire (Schofield et al., 2006), (Schofield and Hägglund, 2008):

$$\left(\frac{F_x^{ij}}{F_{x,max}}\right)^2 + \left(\frac{F_y^{ij}}{F_{y,max}}\right)^2 = 1 \tag{6.11}$$

  The maximum longitudinal and lateral forces of each tire can be computed from the longitudinal and lateral friction coefficients and the vertical load

$$F_{x,max}^{ij} = \mu_x^{ij} F_z^{ij} \tag{6.12}$$

$$F_{y,max}^{ij} = \mu_y^{ij} F_z^{ij} \tag{6.13}$$

Eventually, the Friction ellipse derived bound is given by

$$|T_{max,frict}^{ij}| = R\mu_x^{ij} F_z^{ij} \sqrt{\left(1 - \left(\frac{F_y^{ij}}{\mu_y^{ij} F_z^{ij}}\right)^2\right)} \tag{6.14}$$

6.3.1.1.2 Soft Constraints

On the other hand, when a constraint is soft, the controller can deem a manipulated variable optimal even though it predicts a violation of that constraint. Therefore, when working with soft constraints quadratic programming (QP) infeasibility does not occur. However, the controller can decrease its performance.

Equation (6.4g) is allowed to be slightly violated using the slack variable ξ since we want that the applied torques be as closer as possible to the vehicle torque request without overshooting it. This way, a linear penalty term has been added to the objective function to punish large positive values of ξ_T through the hyperparameter λ (see Equation (6.4a)).

Note that if the problem solution is infeasible the constraints in which the variables involved in are not quite crucial, these could be relaxed to try to get a feasible solution.

6.4 Solvers

In this section two different approaches will be considered. Firstly, YALMIP and afterwards, FORCES Pro.

6.4.1 YALMIP

YALMIP (Yet another LMI parser) is a free MATLAB toolbox for rapid prototyping of optimization problems. YALMIP can be used for linear programming (LP), quadratic programming, second order cone programming (SOCP), semidefinite programming (SDP), mixed-integer programming (MIP), etc (Löfberg, 2004).

YALMIP is a high-level modeling language for optimization in MATLAB and allows to write self-documenting code that is expressed following a mathematical description of the optimization model

$$
\begin{aligned}
\min \quad & x_N^\top P x_N + \sum_{k=0}^{N-1} x_i^\top Q x_k + u_k^\top R u_k \\
\text{s.t.} \quad & x_0 = x \\
& x_{k+1} = A x_k + B u_k \\
& \underline{x} \leq x_k \leq \overline{x} \\
& \underline{u} \leq u_k \leq \overline{u}
\end{aligned}
\tag{6.15}
$$

6.4.1.1 YALMIP implementation

In order to work jointly with YALMIP and Simulink, an NMPC object creation step should be done since creating an NMPC object each time does not have sense in terms of computational demand and the meaning of the object as the structure does not vary between optimization steps.

Firstly, the optimizer object is built gathering the optimization aspects, inputs and outputs of the NMPC problem. So, the aim is to pre-initialize the solver offline in order to be called online afterwards inside the Simulink scheme. This object contains the code that carries out the NMPC optimization problem considering the `sdpvar`, YALMIPs symbolic decision variables, that will be modified during the optimization stage; the input parameters that are constant during the prediction horizon but that change between NMPC calls.

Moreover, the loop involving both constraints and objective function is included. Once the problem is defined, several options in the optimizer are set through `sdpsettings` that is used to communicate options to YALMIP and solvers. Here a key aspect is shown, the solver, in this case the IPOPT (Interior Point Optimizer) is used, which is an open-source software package for large-scale nonlinear optimization (COIN-OR). It can be used to solve general nonlinear programming problems of the form

$$
\begin{aligned}
\min_{x \in \mathbb{R}^n} \quad & f(x) \\
\text{s.t.} \quad & g^L \leq g(x) \leq g^U \\
& x^L \leq x \leq x^U
\end{aligned}
\tag{6.16}
$$

The advantage of IPOPT is that it does not require a special structure of the optimization problem and can be called using YALMIP language. In order to simulate the NMPC, both the nonlinear plant and model plant can be used just by commenting out or through the Simulink subsystem.

The NMPC predicted outputs using the Ipopt solver can be seen in Figure 6.1 and 6.2. The values are quite close to the plant but the most detrimental aspect is the slowness when optimizing.

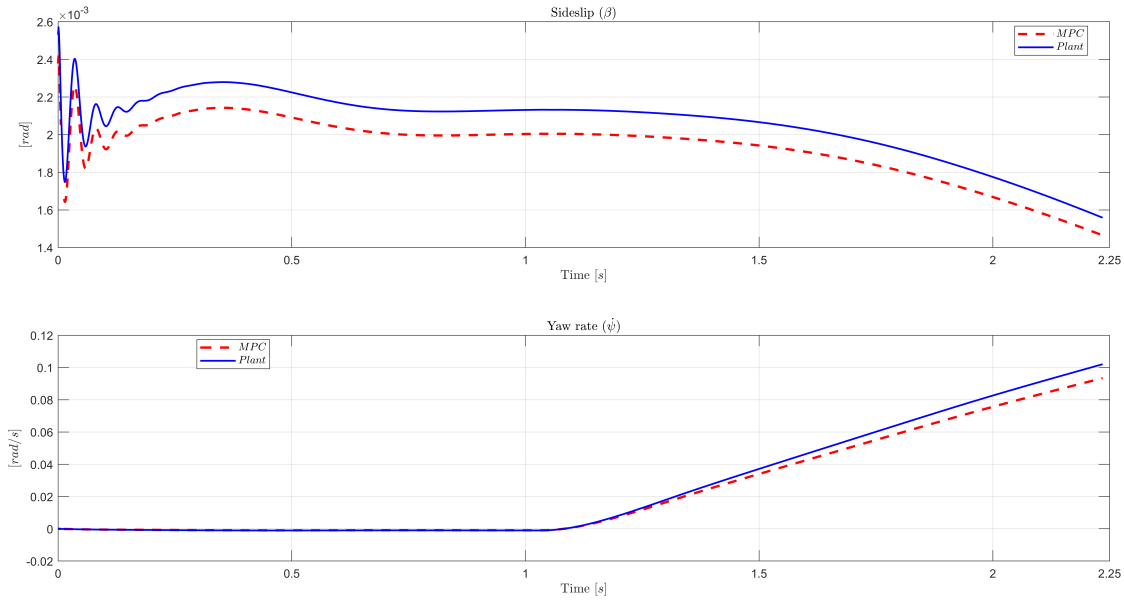


Figure 6.1: MPC predicted states vs Plant model

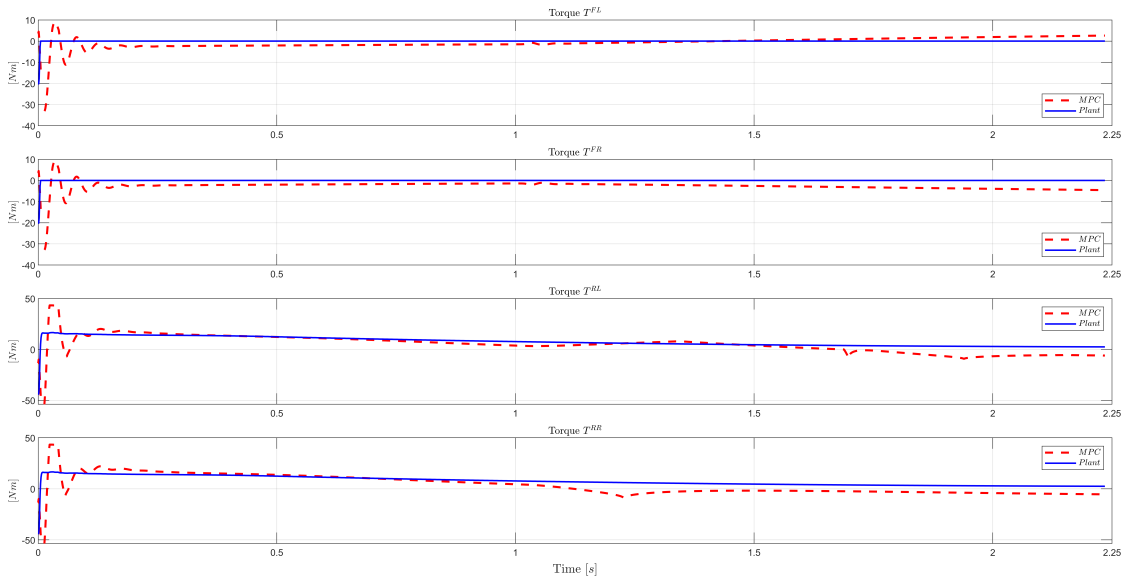


Figure 6.2: MPC predicted torques vs Plant model

Therefore, the NMPC will progressively increase in complexity to verify that the results agree with those of the plant. However, the simulations last long time. A computational efficiency ratio (simulation time elapsed unit per real time unit), $\eta_{sim/real} > 200$ is obtained, which makes it almost impossible to perform debugging due to the time it takes to simulate a small portion of time. So, in this case, IPOPT is not ideal for a

real-time application.

For this reason, instead of spending time tuning the NMPC and analyzing the effects of varying weights, it is better to find an appropriate solver that suits the problem. This has also been found in references where they suggest using more industry-standard solver, more appropriate for this kind of problems (Siampis et al., 2017).

6.4.2 FORCES Pro

After revising the solvers used in the literature, the most efficient ones require a special structure of the optimization problem, so it is necessary to formulate the problem in a powerful solver without computational limitations. This is a real-time NMPC strategy that employs the real-time iteration scheme (Diehl et al., 2002a).

The RTI approach consists in performing the Newton steps always using the latest information on the system evolution (Gros et al., 2016). The RTI scheme is divided into two phases:

- A **preparation phase**, computations prior to obtaining the new state estimate $\hat{x}_i$.
- A **feedback phase**, computations upon obtaining the latest state estimate $\hat{x}_i$.

In (Siampis et al., 2017), the authors address three real-time-implementable NMPC formulations to perform a step steer maneuver whose steering inputs range from 2° to 10° and use different initial velocities. The step steer and other maneuvers involved in the simulation will be described and tested in Chapters 7 and 8. The performance comparison can be seen in Figure 6.3 with the following strategies:

TABLE II
COMPUTATIONAL TIMES AND PERFORMANCE RESULTS FROM THE MPC STRATEGIES

	Avg comp. time (ms)	Max comp. time (ms)	Min per. penalty (%)	Max per. penalty (%)
Linear MPC	1.1	5.3	28.08	109.85
NMPC-RTI	3.0	14.9	2.01	$5.91 \cdot 10^5$
NMPC-PDIP	3.6	29.5	0.79	28.23

Figure 6.3: Linear and nonlinear NMPC performance

1. Linear NMPC: the continuous dynamics are linearized around the equilibrium point(x^*, u^*) and then discretized assuming that the input remains constant for the discretization interval. The formulation can be solved using the PDIP (Primal-Dual Interior-Point) method available in FORCES Pro.

2. NMPC-RTI: it produces fast but suboptimal solutions by precomputing the necessary sensitivities and performing only one SQP (Sequential Quadratic Programming) iteration. This approach can rapidly lead to convergence if the solution does not change much from one time step to the next but it can also diverge.

3. NMPC-PDIP: it attempts to solve the NMPC problem, using the PDIP method, in a relatively short time by employing a Broyden–Fletcher–Goldfarb–Shanno algorithm for the computation of the Hessian of the Lagrangian and can give solutions that are quite close to the optimal.

The Linear NMPC is discarded due to presenting both high minimum and maximum closed-loop costs (expressed as percentage difference from the optimal). As for the nonlinear approaches, there exists a trade-off between problem complexity and performance in order to stay real-time implementable. In addition, FORCES Pro implements both nonlinear versions of previous methods NLSQP and NLPDIP, so it will be very useful to test the NMPC performance.

6.4.2.1 Toolbox capabilities

FORCES Pro is an Embotech's commercial tool for generating highly customized optimization solvers that can be deployed on all embedded computers. FORCES Pro is intended to be used in situations were the same optimization problem has to be solved many times, possibly in real-time, with varying data, i.e. there is sufficient time in the design stage for generating a customized solution for the problem you want to solve (Domahidi and Jerez, 2014–2019), (Zanelli et al., 2017). An overview of FORCES Pro can be seen in Figure 6.4.

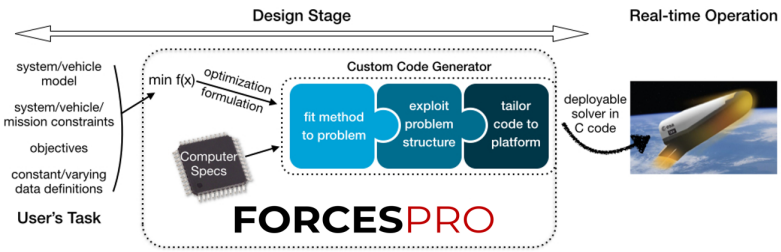


Figure 6.4: Overview of FORCES Pro

The code generation engine in FORCES Pro extracts the structure in your optimization problem and automatically synthesizes a custom optimization solver. The resulting C code can only solve one optimization problem (with certain data changing), hence it is typically many times more efficient and smaller code size than general-purpose optimization solvers. The generated C code is also library-free and uses no dynamic memory

allocation making it suitable for safe deployment on real-time autonomous systems.

6.4.2.2 FORCES Pro formulation

FORCES Pro include several optimization interfaces: YALMIP to FORCES Pro (Y2F), low-level and high-level interfaces. Depending on the problem nature and complexity one of the previous will fit the optimization problem.

- Y2F interface: supports convex decision making problems, with or without binary variables.
- Low-level interface: supports the class of convex multistage quadratically constrained quadratic programming (QCQPs).
- High-level interface: the solver generated from the high-level interface supports nonlinear and convex decision making problems with integer variables. It solves (potentially) non-convex, finite-time nonlinear optimal control problems.

FORCES Pro nonlinear capabilities enable application domain experts to use the power of optimization-based design in very demanding applications without having to worry about complex low-level implementation details. Moreover, for nonlinear problems the tool enables code-generated interior point (IP) and SQP solvers that are not based on finite-difference derivatives computation, resulting in faster convergence (Embotech, 2022). This way, we will be focused on the high-level interface to address our problem which includes both PDIPNLP and SQPNLP.

The FORCES NLP canonical problem for discrete-time dynamics is formulated as follows

$$\min \quad \sum_{k=1}^{N-1} f_k(z_k, p_k) \qquad \text{(separable objective)} \tag{6.17a}$$

$$\text{s.t.} \quad z_1(\mathcal{I}) = z_{init} \qquad \text{(initial equality)} \tag{6.17b}$$

$$E_k z_{k+1} = c_k(z_k, p_k) \qquad \text{(inter-stage equality)} \tag{6.17c}$$

$$z_N(\mathcal{N}) = z_{final} \qquad \text{(final equality)} \tag{6.17d}$$

$$\underline{z}_k \leq z_k \leq \overline{z}_k \qquad \text{(upper-lower bounds)} \tag{6.17e}$$

$$f_k z_k \in [\underline{z}_k, \overline{z}_k] \cap \mathbb{Z} \qquad \text{(integer variables)} \tag{6.17f}$$

$$\underline{h}_k \leq h_k(z_k, p_k) \leq \overline{h}_k \qquad \text{(nonlinear constraints)} \tag{6.17g}$$

$$\forall k = 1, \ldots, N,$$

where $z_k \in \mathbb{N}^{n_k}$ are the optimization variables, i.e. NMPC inputs, states or outputs; $p_k \in \mathbb{N}^{l_k}$ are real-time data; $f_k : \mathbb{R}^{n_k} \times \mathbb{R}^{l_k} \to \mathbb{R}$ are stage cost functions; $c_k : \mathbb{R}^{n_k} \times \mathbb{R}^{l_k} \to \mathbb{R}^{w_k}$ represents (potentially nonlinear) equality constraints, i.e. a state transition function; the matrices E_k are used to couple variables from the $(k+1)^{-th}$ stage to those of stage k through the function c_k; and the function $h_k : \mathbb{R}^{n_k} \times \mathbb{R}^{l_k} \to \mathbb{R}^{m_k}$ used to express potentially nonlinear, non-convex inequality constraints (Embotech, 2022). The index sets $\mathcal{I}$ and $\mathcal{N}$ are used to determine which variables are fixed to initial and final values, respectively. The initial and final values z_{init} and z_{final} can also be changed in real-time. At every stage k, the matrix F_k is a selection matrix that picks some coordinates in vector z_k.

All real-time data is coloured in blue. In addition, when integer variables are modelled, they need to be declared as real-time parameters.

6.4.2.3 FORCES Pro Implementation

As it was done in the YALMIP approach, the idea is to generate a solver offline, in this case, the ForcesPro solver and then call the solver online each time step in Simulink. There are 2 possibilites regarding the ForcesPro call in Simulink:

1. Nonlinear Forces solver block: contains the NMPC built problem generated through the solver option `codeoptions.BuildSimulinkBlock = 0`. The Simulink block can be seen in Figure 6.5 and it allows to connect inputs and outputs via bus in the Simulink interface as if it were a blackbox.

Figure 6.5: NMPC controller Simulink block

2. MEX interface: calling the ForcesPro solver through the MEX file which has been generated by means of the the MEX interface. It is similar to what has been done in the first approach through a Matlab function, enabling to set initial guess to start solver, initial conditions, etc. Note that is crucial to declare the `coder.extrinsic('FORCES_NLP_solver')` extrinsic function in order to allow

Matlab code generation so that the MATLAB engine can execute the call (MathWorks, 2022a).

The advantage of the second calling method is that it allows a customized NMPC problem formulated by the user since any coded optimization problem can be called from script whereas the first method has predefined problem formulation options.

Once the benchmarking in terms of computational speed and the ease of making changes (debugging and tuning) is complete, the module will be prepared to be tested in a real simulator, that is the conversion from the current scenario to a real simulation.

To deploy and simulate a ForcesPro controller on a platform, several steps must be taken into account. Firstly, to generate the necessary files and C libraries, the following code generation options must be set: `codeoptions.platform = '<platform_name>'` to specify the platform in which the solver will be deployed; and `codeoptions.cleanup ... = 0` to keep necessary files for target compile. Then, the solver properties (problem information, outputs, formulation) are saved in structures that will be mapped with C libraries. Once, the linking stage is done, the generated C code from Maltab code allows to make solver compilation in Simulink.

The platform chosen is the Concurrent Real-Time Machine which is a provider of high-performance real-time computer hardware which runs SIMulation Workbench (SimWB). SimWB is the brain software in charge of managing all the simulation process, defining all the signals database and making possible communication between rest of software. Among the process running there is the vehicle model of VI-CarRealTime that defines all vehicle subsystem and components and also the NMPC solver for the motion controller (SimWB, 2022).

6.4.2.3.1 Defining the NMPC problem

For the sake of clarity, let define first the optimization variables, $z_k = (\Delta u_k, u_k, x_k, \xi_T)$ which comprises the torque rates; torques; states (sideslip angle and yaw rate); and the torque slack variable.

A multistage problem structure for $H_p = 5$ is performed in order to generate the FORCES Pro solver. The key aspects will be described as follows:

- When $k = 0$, that is, the first prediction step, different aspects must be taken into account. The initial objective function considers the extra term $J_{\tilde{u}}$ that only applies in this step (input change through optimizer calls). Regarding the inequality constraints, see Equation (6.17g), the input change through optimizer calls $(u_{(k=0)} - u_{-1})$ is also included to smooth the torque variation over time.

- For $k = 1 : (H_p - 1)$, the extra objective term considered for $k = 0$ is now discarded as well as in the inequality constraint term, leading to compute the same objective function and constraints that will hold during the prediction horizon.

Regarding the inter-stage equality constraints, in Equation (6.17c), it is crucial to include the internal model of the NMPC. To do this, there are two approaches:

1. Discrete-time: describe the plant equations in a discrete manner $x_{k+1} = f(x_k, u_k, p_k)$ through an anonymous function or a function handle, i.e `model.eq{i} = ... @(x,u,p) discreteDynamics(x,u,p)`.

2. Continuous-time: describe the vehicle model in a continuous manner $\dot{x} = f(x, u, p)$, i.e `model.continuous_dynamics{i} = @(x,u,p) continuousDynamics(x,u,p)` and then discretize the system using an integrator.

The advantage of using the continuous-time formulation lies in having the ability to customize the integrator types and its properties such as the number of intermediate integration nodes within that integration interval. So, it has been decided to model the dynamics in the continuous-time manner and then discretizing it using an integrator. Among the family of Runge-Kutta iterative methods, the one used is the Runge–Kutta fourth-order method (RK4) by which a fourth-order approximation of the solutions of ordinary differential equations is obtained.

The selection matrix E determines which components of the stage variable z_k will be considered as states x or inputs u. So, the selection matrix E will be used to link the stage variables z_k with the states x and inputs u of the continuous dynamics function. The components of z_k are considered as state variables x according to the order prescribed by the selection matrix. Note that if an entire $k - th$ column of E is zero, the $k - th$ component of z_k is not governed by a dynamic equation and thus considered as input u. Following the formulation described for the high-level interface, $E_k z_{k+1} = c_k(z_k, p_k)$, the selection matrix and the equality constraint term are defined

$$E_k = \begin{bmatrix} 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 \end{bmatrix}, z_k = \begin{bmatrix} \Delta T_{FL} \\ \Delta T_{FR} \\ \Delta T_{RL} \\ \Delta T_{RR} \\ T_{FL} \\ T_{FR} \\ T_{RL} \\ T_{RR} \\ \beta \\ \dot{\psi} \\ \xi_T \end{bmatrix}, c_k = \begin{bmatrix} \Delta T_{FL} + T_{FL} \\ \Delta T_{FR} + T_{FR} \\ \Delta T_{RL} + T_{RL} \\ \Delta T_{RR} + T_{RR} \\ \dot{\beta} \\ \ddot{\psi} \end{bmatrix} \tag{6.18}$$

Thus, the high-level formulation inputs $\tilde{U} = \{\Delta T_{FL}, \Delta T_{FR}, \Delta T_{RL}, \Delta T_{RR}, \xi_T\}$ and states $\tilde{X} = \{T_{FL}, T_{FR}, T_{RL}, T_{RR}, \beta, \dot{\psi}\}$ are obtained. This can be clearly seen by looking at the columns of the selection matrix. Note that the input rate change dynamics is defined in the following way to obtain the input rate change over the prediction horizon

$$\begin{bmatrix} 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \boxed{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \cdot \left. \begin{bmatrix} \Delta T_{FL} \\ \Delta T_{FR} \\ \Delta T_{RL} \\ \Delta T_{RR} \\ \boxed{T_{FL}} \\ \boxed{T_{FR}} \\ \boxed{T_{RL}} \\ \boxed{T_{RR}} \\ \beta \\ \dot{\psi} \\ \xi_T \end{bmatrix} \right|_{(k+1)} = \begin{bmatrix} \boxed{\Delta T_{FL} + T_{FL}} \\ \boxed{\Delta T_{FR} + T_{FR}} \\ \boxed{\Delta T_{RL} + T_{RL}} \\ \boxed{\Delta T_{RR} + T_{RR}} \\ \dot{\beta} \\ \ddot{\psi} \end{bmatrix} \tag{6.19}$$

Therefore, the torque rate change is well defined for all the tires. Let isolate the torque rate of the front-left tire, i.e. $1 \cdot T_{k+1}^{FL} = \Delta T_k^{FL} + T_k^{FL} \Rightarrow \Delta T_k^{FL} = T_{k+1}^{FL} - T_k^{FL}$. Additionally, the c_k matrix also contains the dynamic equations of the plant vehicle.

The optimization variables z_k are also bounded and the run-time parameters p_k used in the problem formulation are updated during the prediction horizon via the solver call

each step instance. This is the case of the inequality bounds that are parametric as well.

Another aspect to consider is the possibility of giving initial and final conditions for the optimization variables.

6.4.2.3.2 Generating the FORCES Pro solver

Once the NLP problem is defined for all stages, it is time to set the desired `codeoptions` that the solver will use. The solve method is an option that allows to use whether `'SQP_NLP'` or `'PDIP_NLP'` and their respective specific options. The user can also set several options to speed the solver optimization.

As for the solver initialization, the performance of the solver can be influenced by the way the variables are initialized. Some problems arise due to the fact of using the default method (cold start) involving an unreliable prediction for the yaw rate in the very first simulation step $t_{sim} = 0.01\,s$. Nevertheless, by setting the centered start method (`codeoptions.init = 1`), the output was pretty close to the external plant output. The centered start method sets all primal variables to zero, the slacks to the RHS of the corresponding inequality, and the Lagrange multipliers associated with the inequalities such that the pairwise product between slacks and multipliers is equal to the parameter ($\mu_0 : z_k = 0, s_k = b_{ineq}$ and $s_k \lambda_k = \mu_0$) (Embotech, 2022).

By default, the solver returns the solution vector for all stages as multiple outputs. If a third argument is added to an array that specifies what the solver should output to the solver generation function `solver = FORCES_NLP(model,codeoptions,outputs)`, the solution vector for all stages contains now the desired ordered variables, $\tilde{Y}_{k:M}$

$$\begin{aligned} &\tilde{Y}_{k:M} = (\Delta u_{k:M}, u_{k:M}, x_{k:M}, \xi_{k:M}) \\ &M = N - 1, \\ &\forall k = 0, \ldots, N \end{aligned} \tag{6.20}$$

6.4.2.3.3 Calling FORCES Pro solver

If the code generation step has been successful, the user can call the solver passing a set of arguments to the generated solver. First, it is necessary to allow Matlab code generation (`coder.extrinsic('FORCES_NLP_solver')`).

The ForcesPro NLP solver solves NLPs to local optimality, hence the resulting optimal solution depends on the initialization of the solver. Depending on the rate of change of the plant variables it could be useful to set initial guess to start solver to be in the middle of all bounds.

If there are any initial and/or final conditions on the optimization variables, the solver will expect the corresponding runtime fields. The approach lies in using the value of the previous optimal solution if it is found to start the problem instance. The parametric bounds can be stacked on the same array covering all stages by simply setting the following code-generation option `codeoptions.nlp.stack_parambounds = 1`.

Moreover, the runtime parameters, input real-time data needed, are set for the prediction horizon. Now, the solver can be called with the aforementioned input problem structure `[output,exitflag,info] = FORCES_NLP_solver(problem)`, which returns the values of the actual iteration of the solver `output`; the integer `exitflag` indicating the state of the solution; and additional information about the last iteration `info`: number of iterations that lead to the result, time needed for solve (wall clock time), primal objective, etc.

Chapter 7

Implementation

In this chapter, the implementation details are described as well as delving into the considerations taken into account to develop the subsystems interactions. Additionally, both plant model and NMPC will be validated using data obtained from test maneuvers.

7.1 Simulink scheme

The Simulink scheme shows how the systems and subsystems are interconnected. The top level of this scheme can be seen in Figure 7.1. In fact, in order to select a specific block or model when desired, a variant subsystem technique is used (MathWorks, 2022d). Variant subsystems provide multiple implementations for a subsystem where only one implementation is active during simulation. So, the user can switch among variants without modifying the model. The proposed scheme is divided into five systems which in turn contain several subsystems.

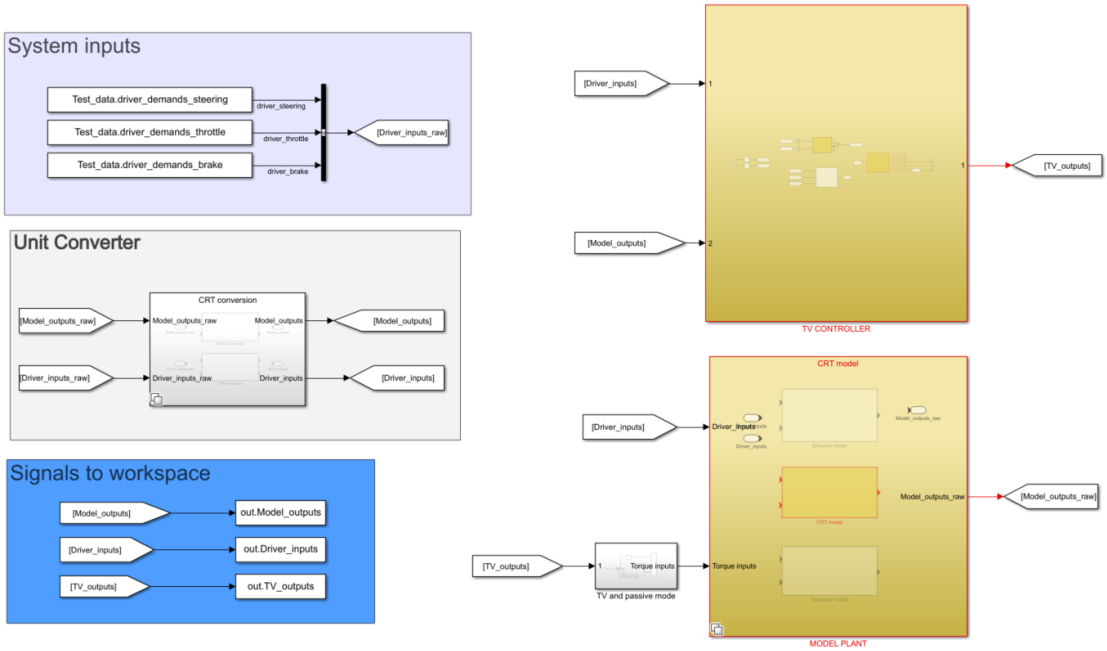


Figure 7.1: Simulink scheme

- System inputs: once a maneuver is designed, the vehicle model is simulated to perform the maneuver. This step concludes with obtaining a set of driver demands (steering angle, throttle, brake) that will be the inputs.
- Unit converter: is the system in charge on carrying out the unit conversion from the vehicle model to the outputs unit. The variant subsystem allows to set either the VI-CarRealTime (CRT) plant or the Simulink plant depending on the plant used to do the simulation. Thus, the `VSS_MODE` variable is set to 1 to use the Simulink plant `VSS_SIMULINK_MODEL = Simulink.Variant('VSS_MODE==1')` and `VSS_MODE` is set to ≥ 2 to use CRT plant for both Simulink simulation and real machine (simulator) simulation `VSS_CRT_MODEL = Simulink.Variant('VSS_MODE≥2')`.
- Torque Vectoring controller: this system is devoted to carry out NMPC controller computations. This module will be described in detail afterwards.
- Model plant: using a variant subsystem architecture, the user can select the vehicle model depending on the platform where the model is running. This selection can be easily made by setting the `VSS_MODE` variable to 1 for the Simulink plant `VSS_SIMULINK_PLANT = Simulink.Variant('VSS_MODE==1')`, 2 for the CRT plant `VSS_CRT_PLANT = Simulink.Variant('VSS_MODE==2')`, or 3 for the simulator plant `VSS_SIMULATOR_PLANT = Simulink.Variant('VSS_MODE==3')`. In addition, it is possible to select the controller activation or deactivation, vehicle

passive mode, by setting the flag `TV_activation` to 1 or 0, respectively.

- Workspace outputs: the system useful variables (signals) are saved in the workspace to do the corresponding system validation and analysis.

7.1.1 Workflow

The signals that intervene in the communication scheme presented in Figure 7.1 will be described.

7.1.1.1 Maneuver design

Firstly, it is crucial to know how the driver demands are obtained. In order to collect the system inputs, a maneuver has to be parameterized and generated. For this purpose, VI-CarRealTime tool will be used. When defining an maneuver event, two control methods are available for each control signal (VI-GRADE, 2021). The control methods are listed below:

- Open: uses open loop control where the control signal is a function of time. Depending on the selected control type (open loop maneuver such as constant steer test), the information to define the shape of the steering control signal must be specified.
- Machine: uses closed loop lateral and longitudinal controllers to follow a path and target velocity or longitudinal acceleration profiles.

Once the input parameters of a maneuver have been set, the maneuver can be simulated running a VirtualTestDrive event in combination with VI-DriveSim (driving simulator). The VirtualTestDrive event is used to generate the input files for the VI-CarRealTime interface with VIRES VirtualTestDrive (VTD). VTD has been developed for the automotive industry as a virtual test environment used for the development of ADAS systems.

The mode of simulation selected can be *Files only* to generate a *"_send_svm.xml"* file that contains the whole vehicle model data or *interactive* that in addition it allows to monitor the vehicle variables involved in the maneuver over time. This can be seen in Figure 7.2 with the VI-Animator module that shows the simulation results at run-time.

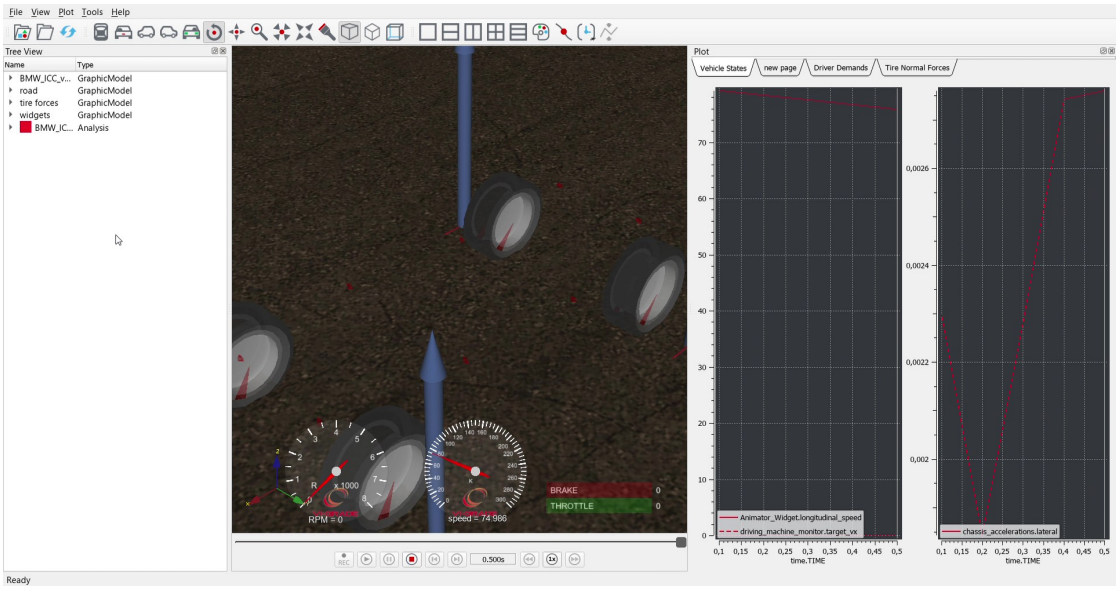


Figure 7.2: Generating a maneuver (VI-Animator)

Once the *"_send_svm.xml"* file has been generated for a given maneuver, the driver demands can be obtained by running the Matlab interface of the VI-CarRealTime solver (CRT plant model). The CRT block can be seen in Figure 7.3. This execution can also be monitored by enabling live animation of the CRT plant by means of VI-Animator. The output bus can export any available model output. However, in this case only the driver steering, throttle and brake signals will be considered.

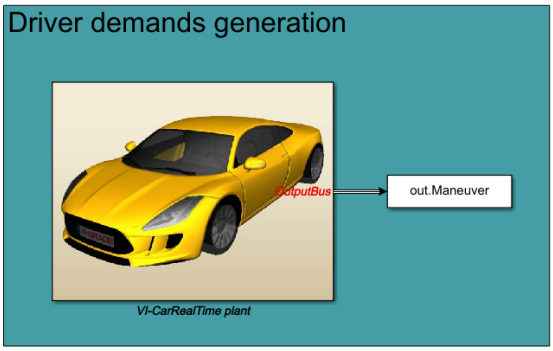


Figure 7.3: Driver demands generation through CRT plant

On the other hand, in order to carry out closed loop maneuvers the Press Maneuvers event will be used. The aim is to use VI-Driver to have the vehicle going through a path while dodging cones or obstacles, at maximum speed available. In this set of maneuvers, a specific trajectory tracking is not done but genetic algorithm is used to generate target trajectories which will be submitted and verified during the dynamic integration, having a constant vehicle velocity as longitudinal dynamics target.

The genetic algorithm comprises 2 phases:

- *Learning*: data is collected from vehicle dynamics.
- *Adapting*: previous trial information data is used to both update trajectory and increase speed.

Thus, depending on the specific maneuver designed, cones will be differently placed in the path. For each iteration, cone interference check is used to verify if the target trajectory is feasible. Once the full set of trajectories have been verified, velocity is increased and a new cycle begins.

The event is completed when a single combination of trajectory and maximum velocity is identified featuring a no cone hit situation. In Figure 7.4, a screenshot of a closed loop maneuver in the VI-Animator is shown. Eventually, to run a co-simulation using a Press-Maneuvers event, the Simulink model must be instrumented using a "PressManeuvers interface" block. This enables that the VI-Driver optimization procedure interacts with the Simulink environment using callback functions, generated by the PressManeuvers interface block, that are called at each iteration.

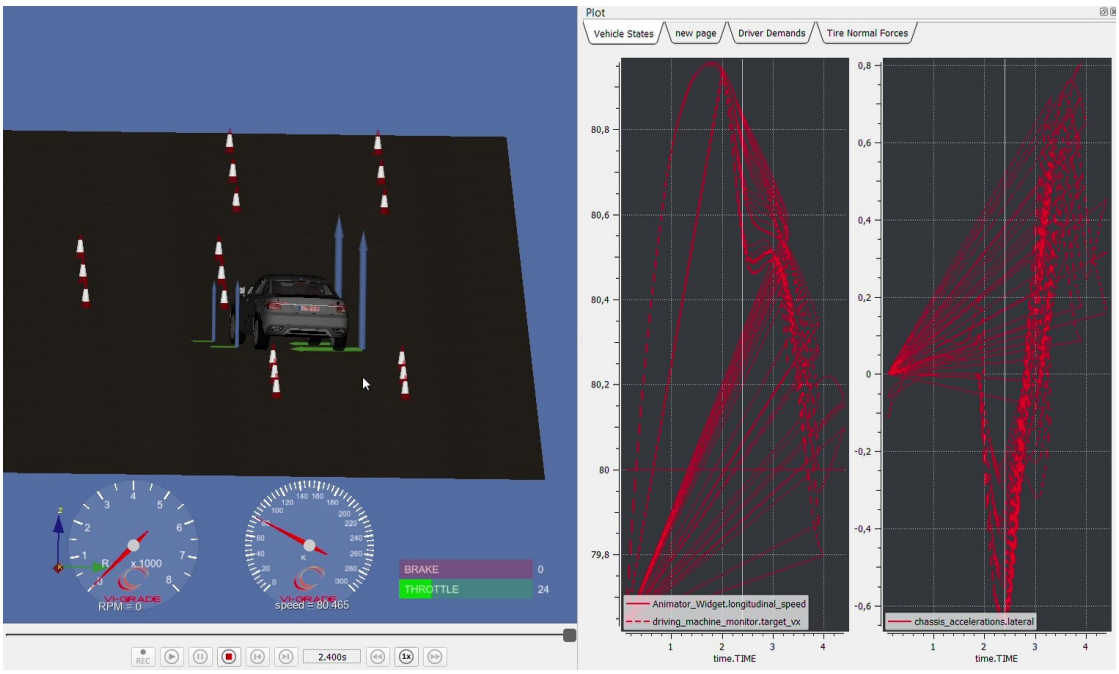


Figure 7.4: Closed loop maneuver (VI-Animator)

In order to ease the understanding of the workflow, the Model plant and the Torque Vectoring controller systems will be analyzed individually.

7.1.1.2 Model plant system

At every iteration, the current plant signals are updated with the input signals that the plant receives (see Figure 7.5). The input information can be divided into different groups according to the nature of the signals:

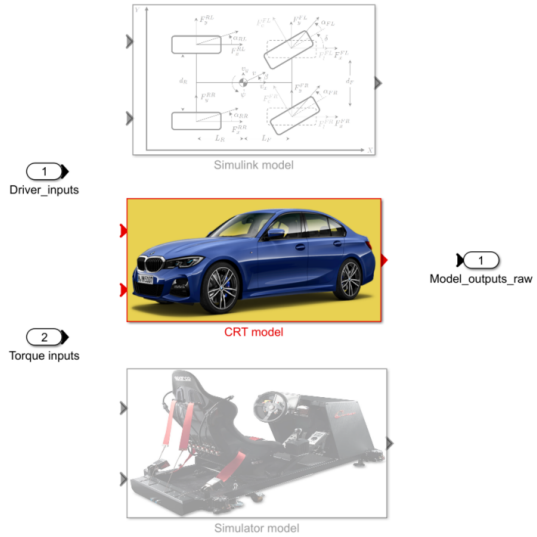


Figure 7.5: Model plant scheme

- From the Driver inputs, brake signal and the driver steering (steering angle, δ, through the steering ratio for the Simulink plant and the SWA for the other CRT plants).
- From the TV outputs, where the NMPC outputs including the torques and states. In case the CRT plant or the simulator plant is selected, additionally, the user can select how to control the engine through the control mode using either torques or throttle; and, using the powertrain map (pedal map) or an external powertrain for the internal motor connected with each wheel.

Once, the model outputs are obtained after each iteration, they enter the Unit converter system to get the appropriate units for further visualization and analysis.

7.1.1.3 Torque Vectoring system

Every 10 iterations ($10\,ms$), the controller acts receiving data from the NMPC input parameters. The controller system is shown in Figure 7.6, the inputs are provided by:

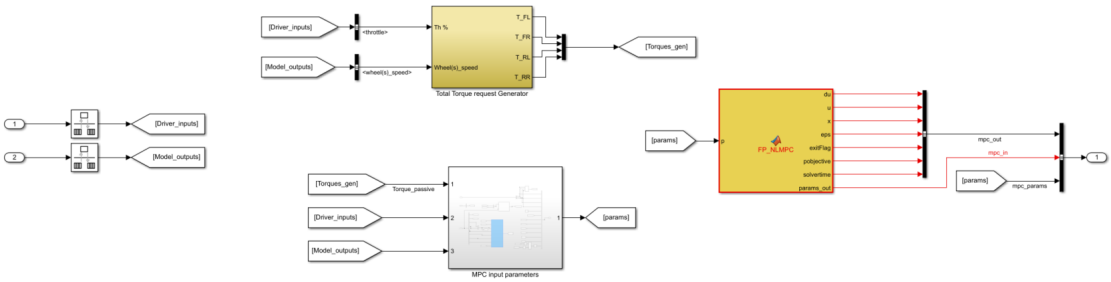


Figure 7.6: Controller scheme

- Driver inputs: considering the driver throttle for the Torque Request Generator in addition to the driver steering for the NMPC input parameters system.
- Torque Request Generator: where the torque request is obtained for each wheel by means of each pedal map.
- NMPC input parameters system: where the online parameters are computed and updated to be used by the NMPC during the prediction horizon. In this module, the yaw rate reference generator; the sum of the individual passive torque signals are added to produce the torque request; the Pacejka coefficients among other parameters are included in a bus signal that enters the NMPC.

On the other hand, the controller provides several signals once each execution is done, such as, the optimized torques that will be the inputs for the vehicle plant. Moreover, the solver provides information about each computation (number of iterations, time needed for solve, exit flag) that are very useful to check the behaviour of the controller.

7.2 Plant validation

Before validating the NMPC controller, it is necessary to ensure that the non-linear plant model behaves similarly to the dynamics given by the BMW 320 dA reference model described in VI-CarRealTime. Once this procedure is fulfilled, the NMPC performance can be analyzed.

Several physical magnitudes and their relationships have to be analysed over time and correlated with the CRT plant. The non-linear plant model will reproduce the behaviour of the CRT model by using the tire model under the tire model parameter *USE_MODE 13* in which the model have pure longitudinal and pure lateral slips, so they are uncombined forces/moment $(F_x, F_y, M_x, M_y, M_z)$.

A crucial requirement of the maneuvers design is to have an open-loop structure which means that not only the sequence of inputs is predefined but also that this sequence is not

affected by the vehicle outputs. Thus, allowing further analysis of the vehicle performance by not using control feedback and also, ensuring test repeatability.

7.2.1 Ramp steer maneuver

Firstly, the ramp steer maneuver, which is a lateral dynamics quasi-static typical test, as an exhaustive test will be performed (Perinciolo and Sondhi, 2018). The ramp steer maneuver is conducted by slowly, linearly increasing the steering wheel input angle, from the centered steering wheel angle, that is, 0° until a given final value. This test will be carried out at 80 km/h and 120 km/h to replicate driving on a country or an interurban road, and on a highway.

A key aspect of this test is the speed at which the steering input varies. There are two limit scenarios that can be analyzed. A low steering rate approximates better steady state cornering conditions, but it takes the car to travel longer and therefore, the maneuver takes longer, which may lead to a space constrain violation in case of physical testing in a proving ground. A high steering rate would fit better when the space constraints are tight, although it implies less accuracy. In order to be in an intermediate situation, the proposed actuation rate is 12 deg/s which is a similar value to the recommended one in the ISO standard describing how to perform this test type in real testing. In Figure 7.7, the steering wheel input for ramp steer is shown.

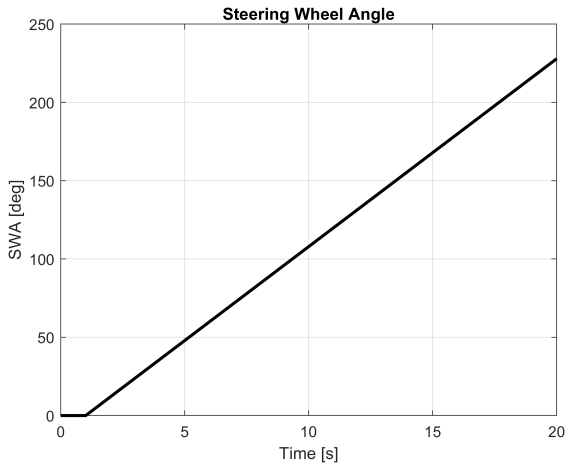


Figure 7.7: Ramp steer maneuver

7.2.1.1 Model variables

To validate the behavior of the plant, a correlation of this model will be made with the CRT plant, which will be treated as the reference model. Some of the key variables are shown in Figure 7.8. The following plots are carried out at a longitudinal velocity that is around 80 km/h.

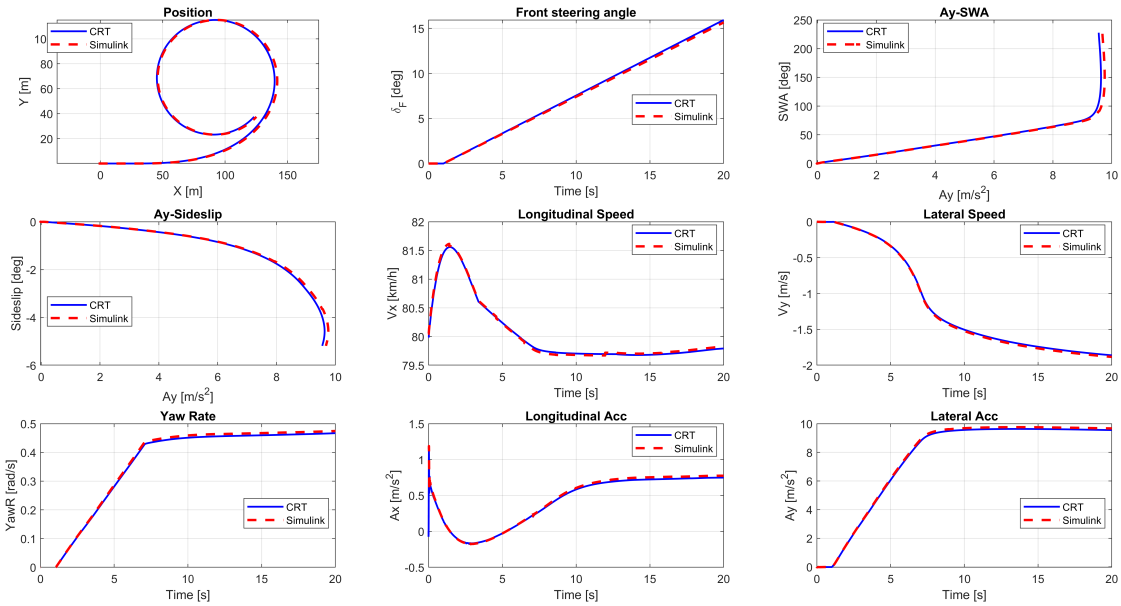


Figure 7.8: Validation of model variables (ramp steer)

The model outputs match at a great extent the CRT plant values. Regarding the yaw rate response, the implemented model does not comprise all the terms taken into account in the simulated model, so it will lead to a certain difference when working in limit handling situation (high lateral acceleration situations when nonlinearities of the tires play an important role, including self-aligning torque) which in any case does not affect heavily the response of the model. Taking a closer look at the graph in the upper left corner, it can be seen that the trajectory obtained by the Simulink plant is very similar to that of the CRT plant.

The evolution of the longitudinal, lateral and normal forces is shown in Figure 7.9. It must be said that there exists a slight difference in the longitudinal forces when there is a sudden change in the CRT plant signal due to the longitudinal force computation, as it considers only the wheel torques and the effective radius of each wheel. A better fit would be feasible using the MF or the equilibrium of forces equation along the x-axis, but this would increase the computational demand. Moreover, both lateral and normal forces presents a good fitting over time.

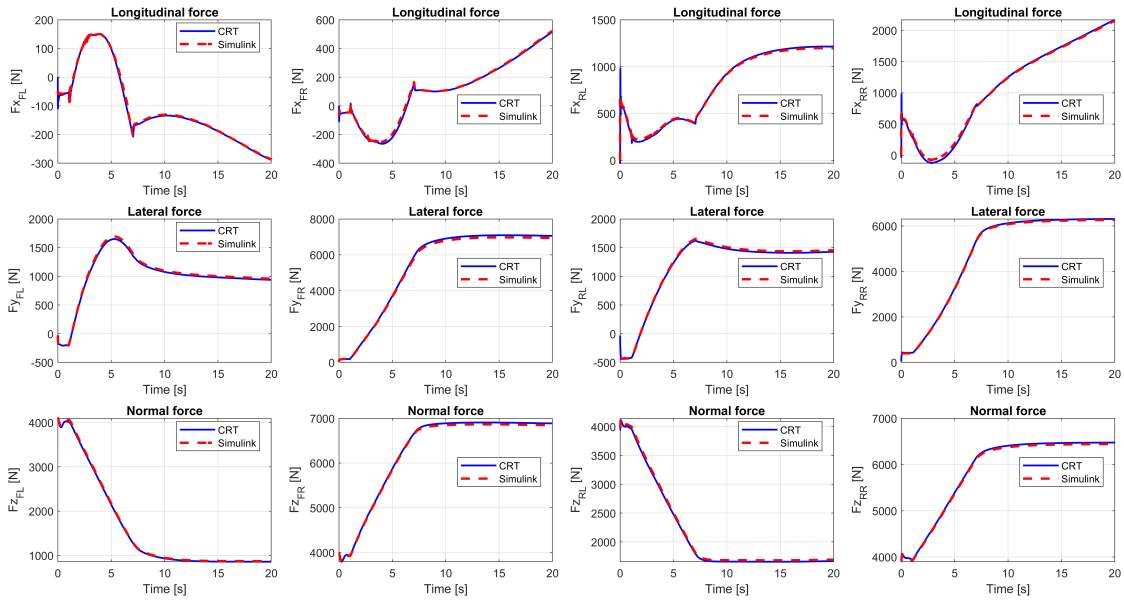


Figure 7.9: Tire forces (ramp steer)

Each engine torque is shown in Figure 7.10. The signals represented are the CRT torque (the reference for correlation) and the Simulink torque, obtained through the engine torque LookUp table from two inputs, throttle and engine angular speed. The results in the front axle are not as accurate as possible while in the rear it works relatively well, since the torque does not consider inertia terms but it becomes a very straightforward and a fast computing approach.

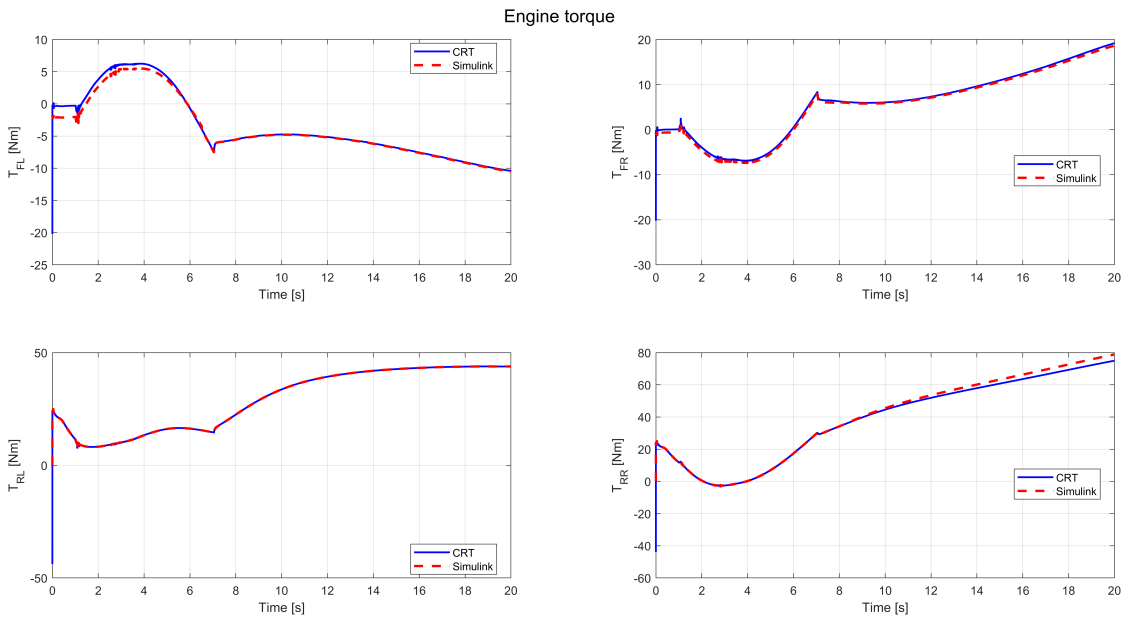


Figure 7.10: Engine torque

The sideslip angle of each tire has been validated in Figure 7.11 to assure a good

model performance. There is only a slight deviation from the CRT signal for the right rear sideslip angle. The rest of the sideslip angles match at a great extent the CRT signals.

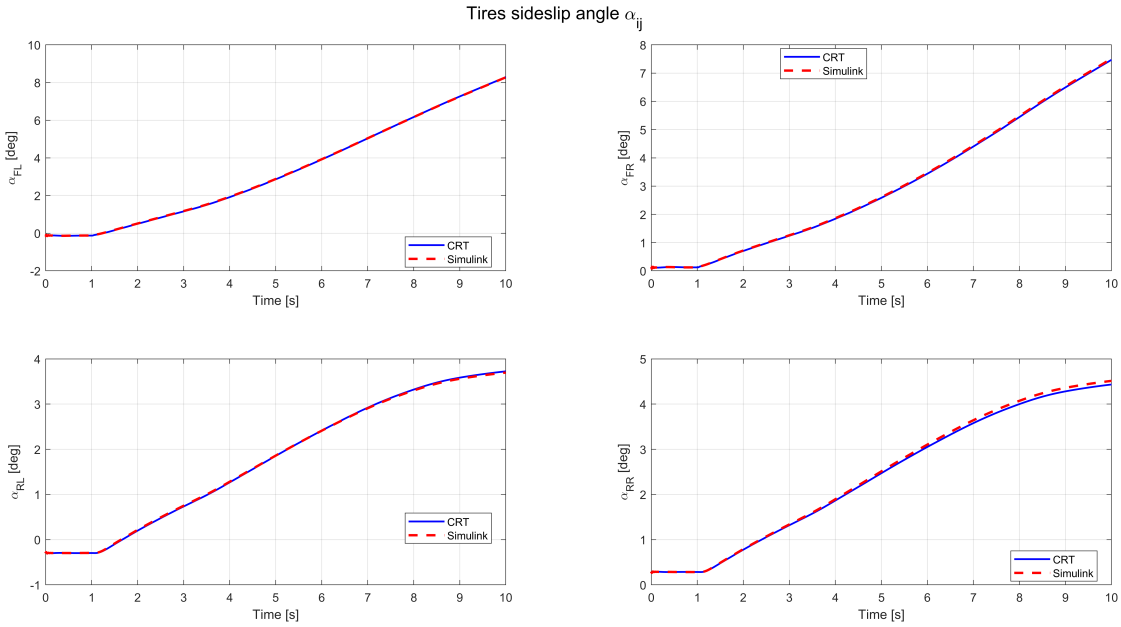


Figure 7.11: Tire sideslip angles

The model validation step has led to similar results using both longitudinal velocities (80 km/h and 120 km/h). However, the model validation through a unique maneuver is not enough, so two additional maneuvers will be performed to extend the correlation with the CRT model.

7.2.2 Step steer maneuver

Now, a dynamics test will be performed. This is the case of the step steer maneuver. This test is quite useful to analyze both transient and steady-state responses. This maneuver consists of changing the steering demand oftenly. A quick sinusoidal steer is performed so that it reproduces an ideal step input, during a given duration to set a final value for the steering wheel angle, which influences the lateral acceleration in steady-state conditions. In order to avoid instrumentation noise and external disturbances the lateral acceleration values cannot be small. On the other hand, excessive values should be discarded to remain in the linear response range (Perinciolo and Sondhi, 2018).

In Figure 7.12, the steering wheel input for step steer is shown. The step duration is 1 second and SWA goes from an angle of 0º to 30º. The test is performed for 80 km/h and 120 km/h.

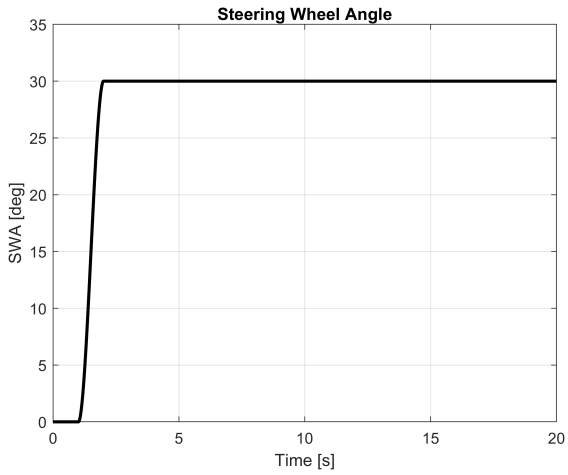


Figure 7.12: Step steer maneuver

Some of the model variables can be seen in Figure 7.13. The lateral acceleration is around 4 $\frac{m}{s^2}$ when it reaches the steady-state. The error in the trajectory travelled by the vehicle is negligible since the difference between the CRT yaw rate and the plant yaw rate is scarce. The overall evolution is satisfactory, so the tire forces can be analyzed.

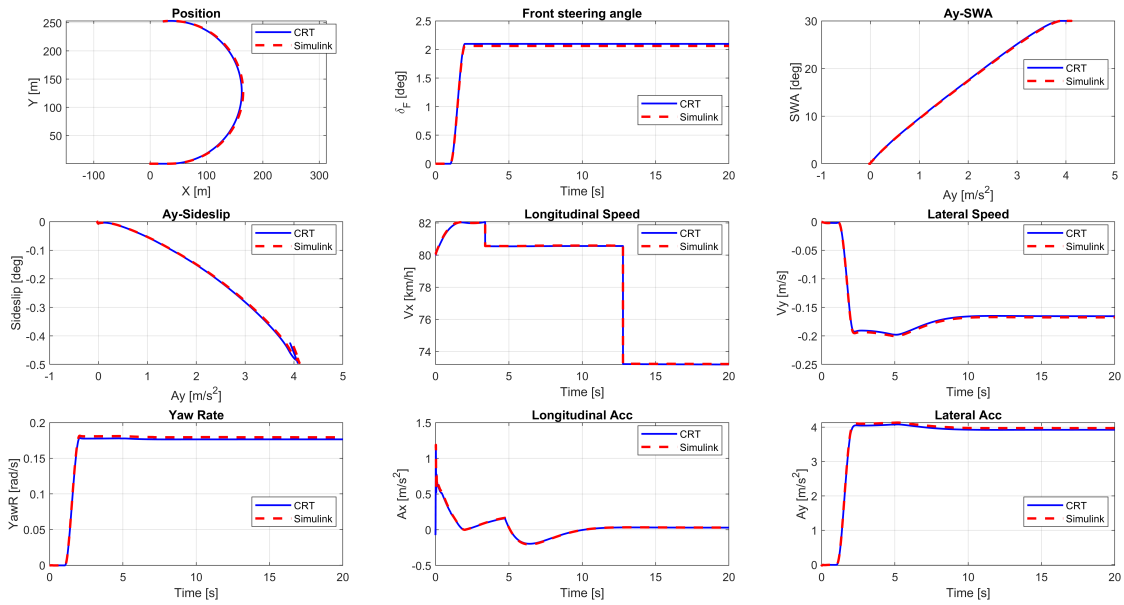


Figure 7.13: Validation of model variables (step steer)

The tire forces are shown in Figure 7.14, in which the correlation is successful. As mentioned in the ramp steer maneuver, slightly differences may appear when a noticeable change in a force is produced. This aspect can be clearly seen in normal force of the rear left and right wheels around 1.5 and 4.8 seconds, respectively.

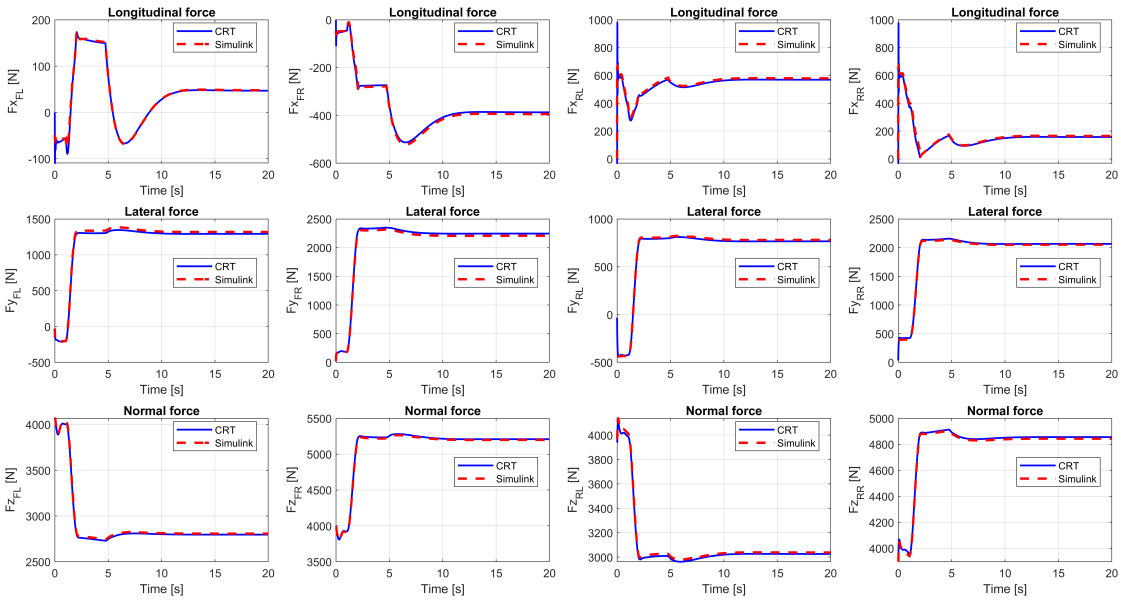


Figure 7.14: Tire forces (step steer)

The model validation step for the step steer maneuver has led to very similar results using both longitudinal speeds (80 km/h and 120 km/h), so no more simulations at different speeds are needed.

7.2.3 Frequency sweep maneuver

Eventually, the swept-sine steer maneuver is performed. This test aims to carry out a frequency sweep, giving a sinusoidal steering demand beginning with a certain amplitude and a sequence of frequencies that starts at a minimum value and goes to a maximum value at given frequency rate. Nowadays, simulations show that modern passenger vehicles show appreciable sensitivity to parameter variation, only inside the interval of 0.2 Hz – 3 Hz. The lower limit describes the intermediate situation between quasi-static and dynamic responses (Perinciolo and Sondhi, 2018). The proposed test lies in a high frequency sweep which covers the range from 0.9 Hz to 4 Hz with a frequency rate of $0.3\,\frac{\text{Hz}}{s}$ and an amplitude of 20° which should correspond approximately to a value of $0.2\,\frac{\text{m}}{s^2}$ in steady-state conditions. The steering wheel angle can be seen in Figure 7.12.

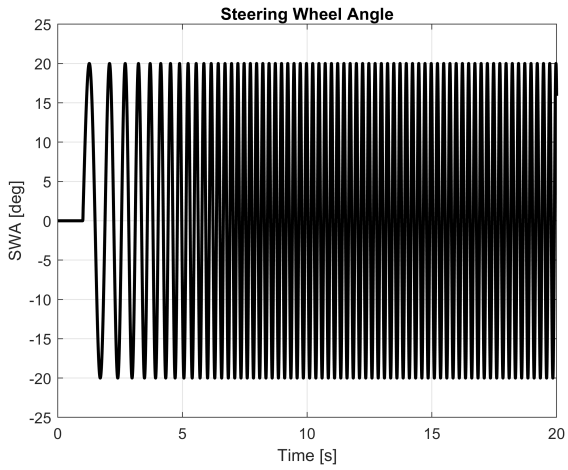


Figure 7.15: Frequency sweep maneuver

The model variables for the swept-sine steer maneuver can be seen in Figure 7.16. There is no significant difference between the phase of the CRT plant and the Simulink plant. There is also no deviation in amplitude, with a difference of less than 1.8 % showing in the peaks of the signal. At $t = 12.8\,s$ the longitudinal speed starts to oscillate considerably since to the input steering angle has already reached 4 Hz.

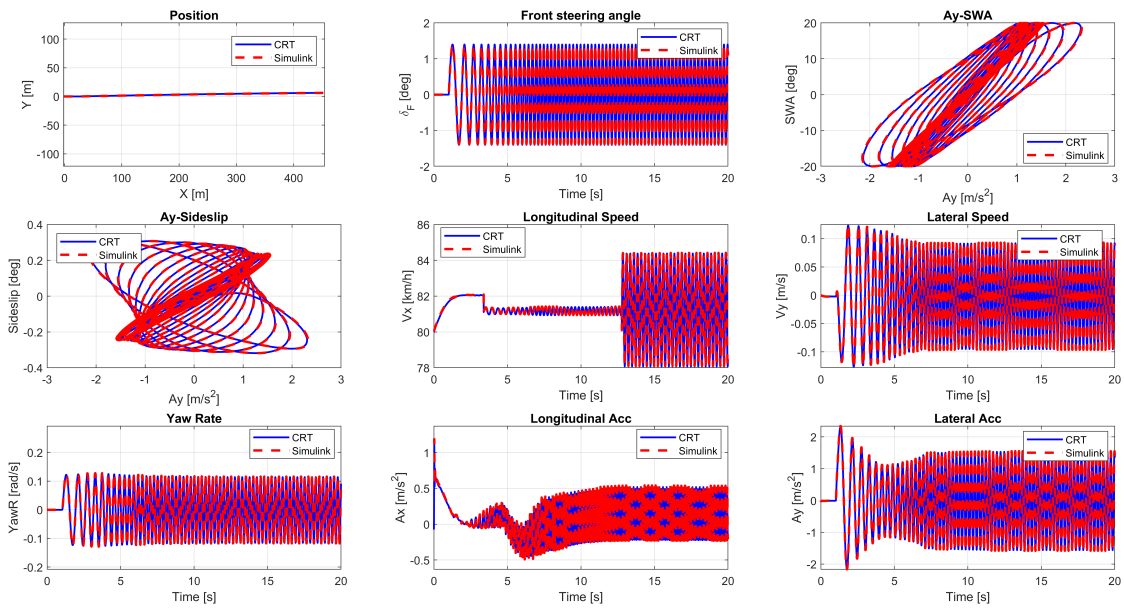


Figure 7.16: Validation of model variables (frequency sweep)

Later, the tire forces evolution is shown in Figure 7.17. As mentioned before, no remarkable differences between CRT and the implemented model are found.

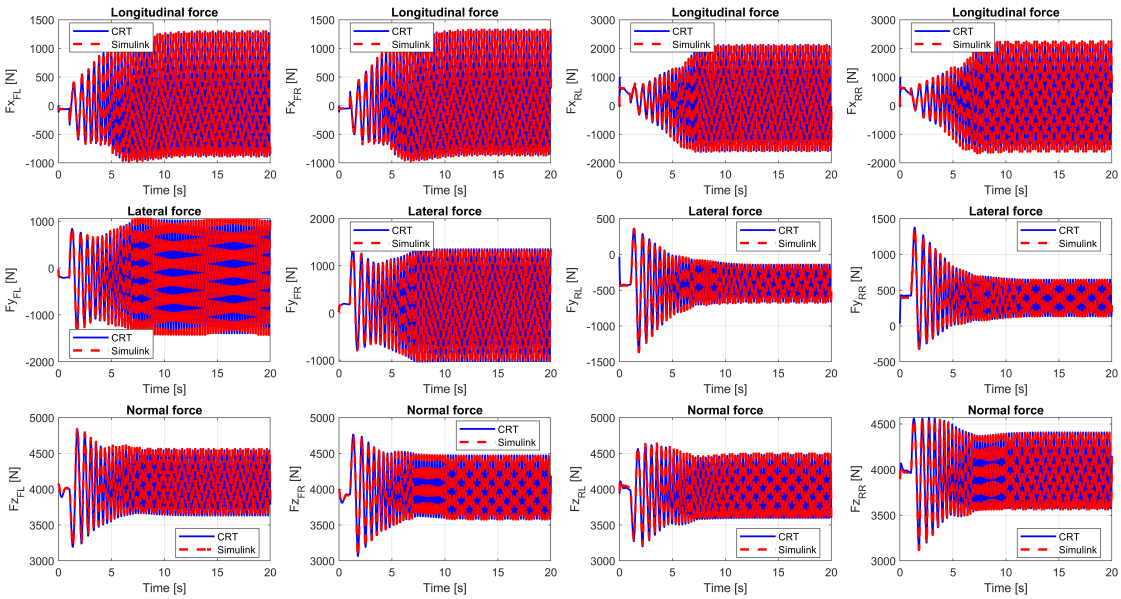


Figure 7.17: Tire forces (frequency sweep)

The model validation step for the swept-sine steer has led to very similar results using both longitudinal speeds (80 km/h and 120 km/h). Therefore, no more simulations will be performed at different speeds.

7.3 NMPC validation

Once the validation stage has been fully covered by testing various maneuvers with different specific objectives, the next immediate step to achieve full validation is the controller validation. Before entering into the NMPC outputs analysis, the tuning procedure is described.

7.3.1 NMPC tuning

A key idea behind of the NMPC algorithm lies in spending time tuning the controller just once, because same or similar tuning works decently in a wide variety of situations. This way, the tuning complexity decreases since no great tuning efforts should be done for each maneuver.

The tuning procedure consists of setting a certain value to the weight matrices found in the objective function so that all cost terms contribute to the optimization problem. First, the terms that play highlighted roles are prioritized by giving them a higher weight or penalization than the other terms. Therefore, after setting these weights only minimal adjustments are needed to further improve the performance.

The methodology starts by a giving a weight to the yaw rate error, Q_{e_x}, and analyze the performance of the controller. After several trials, finding an appropriate weight according to the results obtained, the first weight is set. Then, this procedure is reproduced for the torque error, Q_{e_u}, until reaching the desired weight. Later, the weight to smooth the input change between successive MPC calls, $Q_{e_{\tilde{u}}}$, is chosen. This sequence is repeated until the last weight is set. As mentioned before, slightly tuning modifications can be studied in order to improve the controller performance.

The diagonal values of the weighting matrices (identity matrices), adjusted to obtain the best trade-off among the different objective terms (except Q_{r_u} whose elements on the diagonal are not 1), are

$$\begin{aligned}
Q_{e_x} &= 100 \\
Q_{r_u} &= \begin{bmatrix} 0.35 & 0 & 0 & 0 \\ 0 & 0.35 & 0 & 0 \\ 0 & 0 & 0.65 & 0 \\ 0 & 0 & 0 & 0.65 \end{bmatrix} \cdot Q^*_{r_u}, \quad Q^*_{r_u} = 0.1 \\
Q_{e_u} &= \begin{bmatrix} 1.5 & 1.5 & 1.5 & 1.5 \end{bmatrix} \\
Q_{e_\xi} &= 0.75 \\
Q_{e_{\tilde{u}}} &= \begin{bmatrix} 10 & 10 & 10 & 10 \end{bmatrix}
\end{aligned} \tag{7.1}$$

7.3.2 Ramp steer test

The set of adjusted weights will be used for the different maneuvers performed. Now, the ramp steer maneuver results will be analyzed. The NMPC outputs analysis of the other maneuvers will be described in Chapter 8. The cost of different objective terms per each step of the prediction horizon can be seen in Figure 7.18, where the different cost terms do contribute to the optimization in each step, except the input change which only acts in the first step to smooth the difference between the optimized control action and the previous control applied.

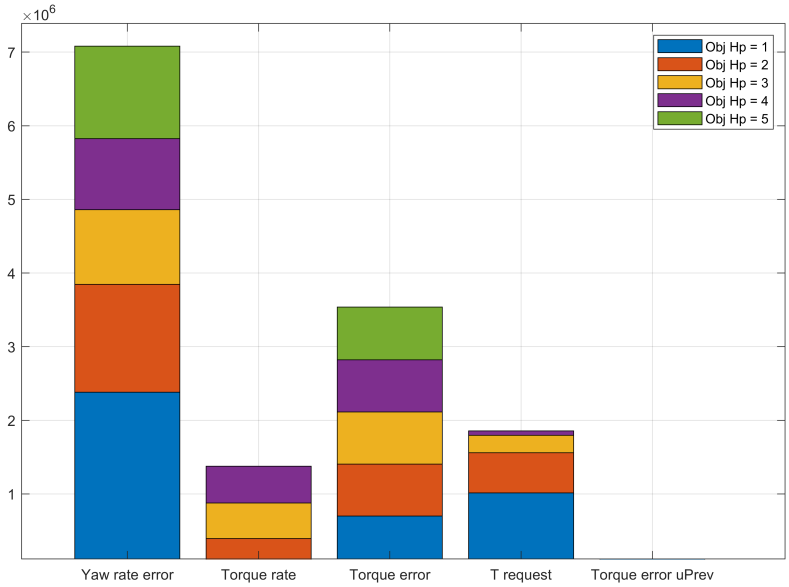


Figure 7.18: Cost of the objective terms per stage

Before analyzing the states, the inputs evolution will be discussed. In Figure 7.19, the optimized torque applied to each wheel, the wheel torque provided by the CRT model and the passive torque reference (torque request) are shown, as well as the torque rate throughout the duration of the maneuver. The upper and lower bounds of the optimization variables (T and ΔT) are also shown with a dotted line in each subplot.

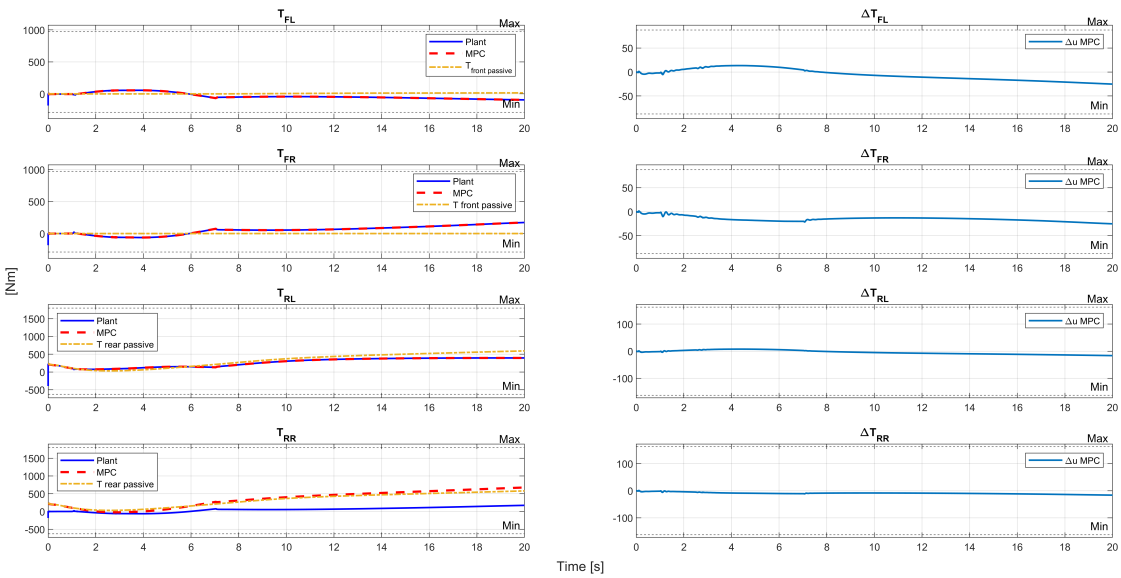


Figure 7.19: Wheel torque and torque rate

The torque rate does not reach the bounds and the signal is smooth without presenting undesirable peaks. Regarding the wheel torques, the torque outputs match the CRT plant values except for the rear right wheel since the torque applied by the NMPC is higher

than the CRT model to improve cornering which leads to maximize torque usage on wheel motors while improving handling capabilities. When cornering, the outer wheels have more capacity to transmit power to the ground than the inner ones, which are more unloaded. Thanks to the selective distribution of torque at the wheels, the change of direction in the car's trajectory can be favored.

In this maneuver, the car is cornering to the left, so the right wheels will have more impact on torque distribution. Anyway, the wheel torque evolution can be seen in a clearer manner in Figure 7.20, all along with the total torque request (sum of the torque request of each wheel) and the sum of the NMPC torques.

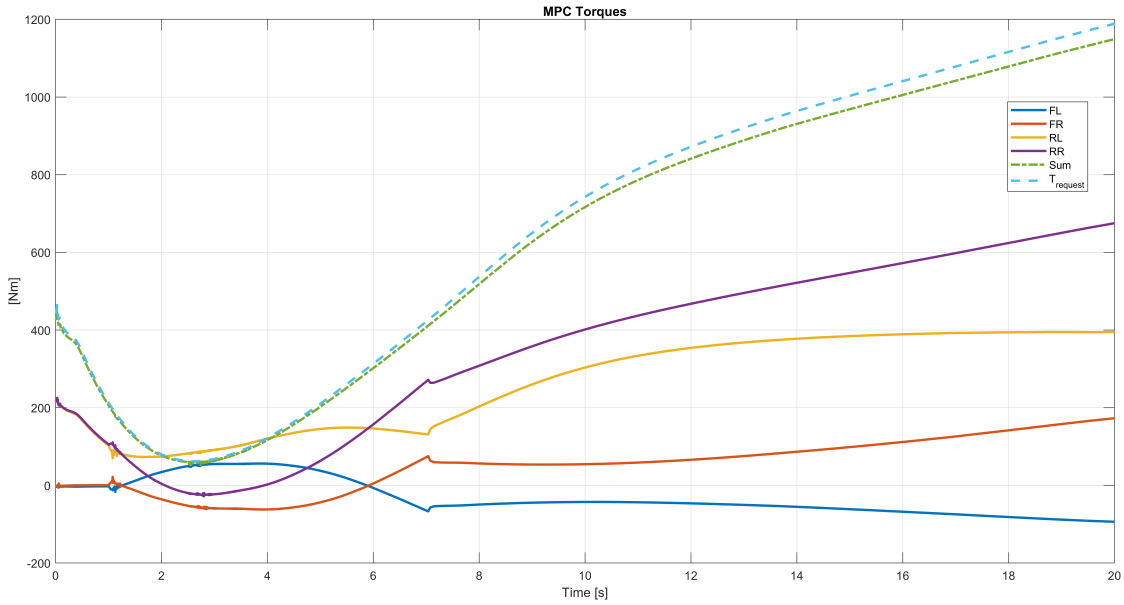


Figure 7.20: Wheel torque distribution

As mentioned, the outer wheels transmit more torque, which allows for more grip, faster cornering, and reduced understeer. The sum of all wheel torques does not reach the total torque request but remains close to it, meaning that the car is close enough to the torque reference without exceeding it. Moreover, it can be seen that the torque distribution varies over time as the vehicle moves forward.

The vehicle states, yaw rate and sideslip angle, are shown in Figure 7.21. The sideslip angle matches at a great extent the CRT plant values. The yaw rate also presents a very similar behaviour to that of the CRT plant. Both NMPC outputs are within the bounds that are plotted (dotted lines) in each subplot.

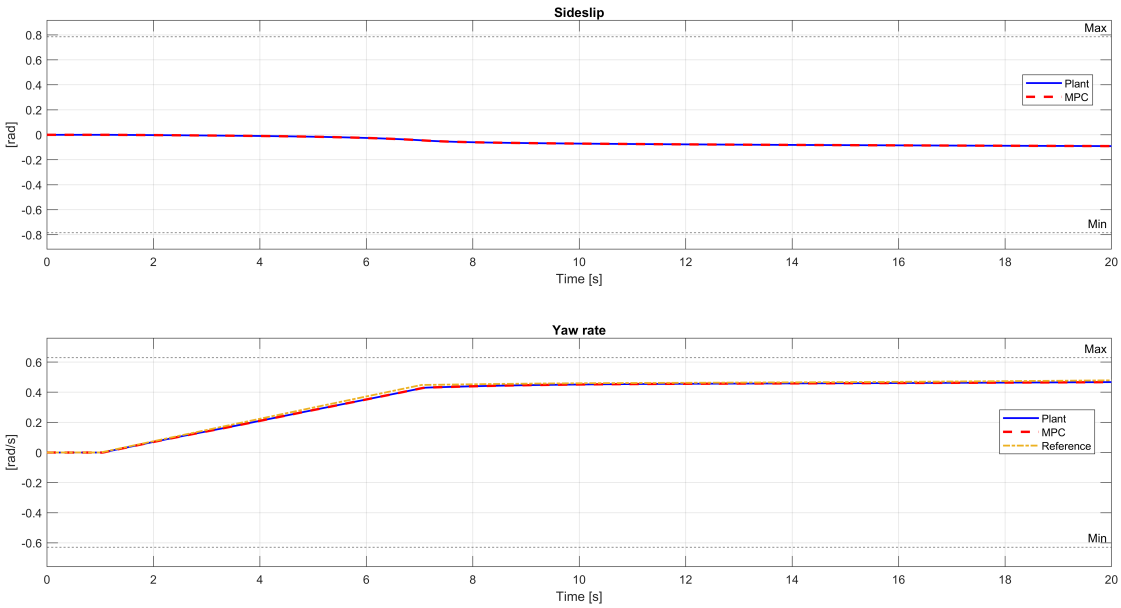


Figure 7.21: States evolution

The yaw rate obtained through optimization is below the yaw rate reference provided by the yaw rate reference generator. The CRT plant is also unable to track the reference yaw rate. This means that the vehicle will not be able to turn as much as the reference value, so there will be a small error in the path traveled.

In addition, some information provided by the SQP solver is shown in Figure 7.22. The exit flag shows that an optimal solution has been found for each optimizer execution throughout the maneuver. The time needed to reach the optimal solution is below the sampling time, which means that the solver obtains a result quicker than the time limit, so the real-time constraint is fulfilled. Subsequently, the cost of the objective function is shown over time. In fact, at time $t \approx 7\,s$, the maximum yaw rate error shows up that coincides with the moment when the yaw rate reference starts to flatten.

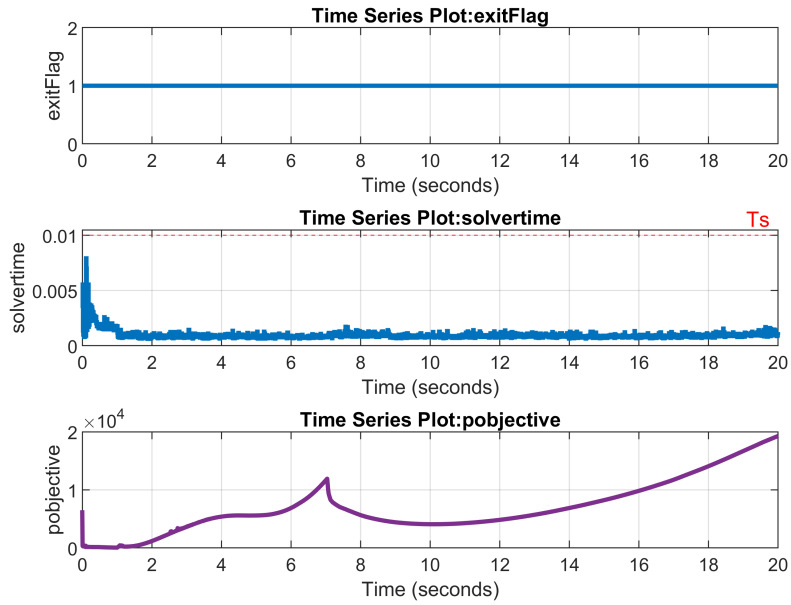
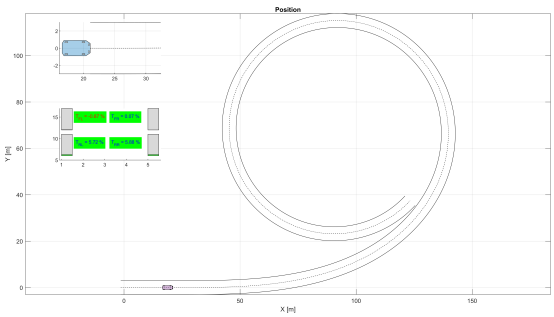


Figure 7.22: Solver information

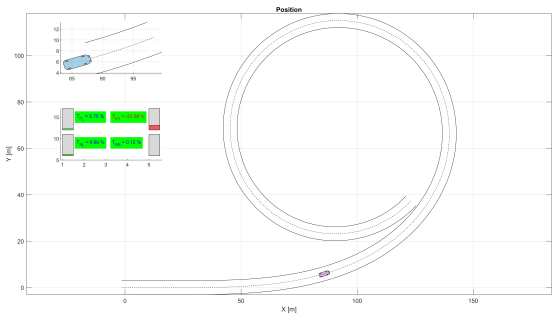
7.3.3 Maneuver trajectory

The vehicle performance under the ramp steer maneuver is shown in Figure 7.23. In the subplots, the path to be followed by the vehicle is represented by a dotted line and there are also track limits located at 1.5 m of the center. In the upper left corner of the figure, a zoomed portion of the current pose of the vehicle is shown. Just below this plot, four wheels are depicted to show the current torque applied. It can be either positive (in green) or negative, that is, braking (in red). The percentage of torque with respect to both negative and positive torque limits given by the pedal map is also shown for each wheel. For instance, a wheel that is completely green means that the current torque applied to this wheel is the maximum. Note that positive and negative torque limits do not coincide, so although the scale (0 - 100) % remains the same, the color height (level) of the wheel drawn correspond to a different torque for negative and positive values, being lower for the negative ones.

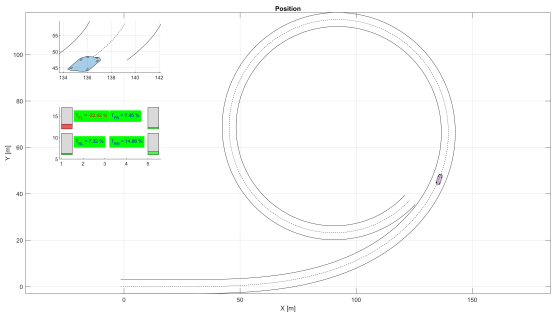
In Subfigure 7.23c, it can be seen that the front left torque is negative and the rear right is positive (higher than in other wheels) to ease the turning of the car while cornering. On the other hand, in Subfigure 7.23f, the vehicle is not completely aligned with the center of trajectory. This orientation difference comes from the yaw rate error as mentioned before.



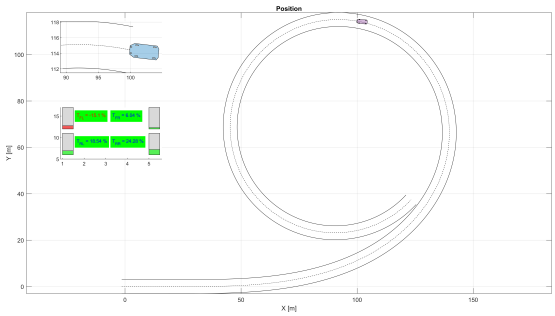
(a) Maneuver sequence 1/6



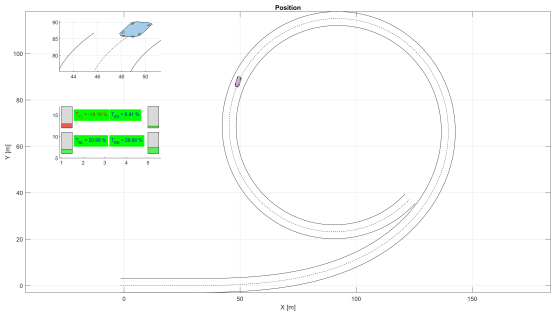
(b) Maneuver sequence 2/6



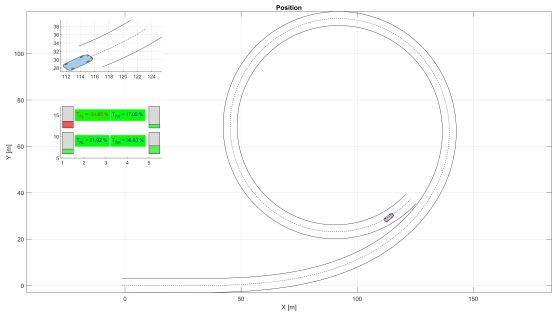
(c) Maneuver sequence 3/6



(d) Maneuver sequence 4/6



(e) Maneuver sequence 5/6



(f) Maneuver sequence 6/6

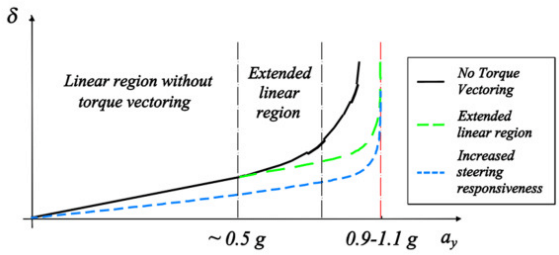
Figure 7.23: Ramp steer maneuver trajectory

7.3.4 Passive mode vs NMPC

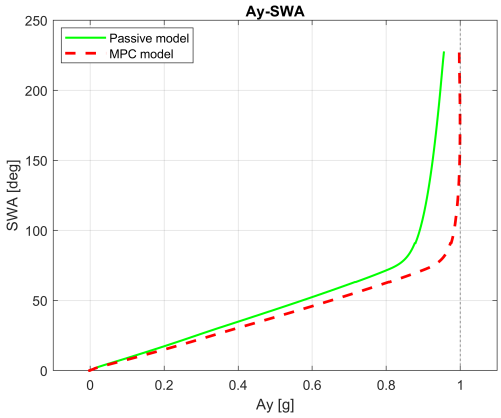
Another interesting comparison to extend the NMPC validation is done for the passive mode of the vehicle which does not use any controller. Vehicle steering characteristic curve can be represented as the steering wheel angle as a function of the vehicle lateral acceleration. The lateral acceleration trend in vehicles that have an understeer behavior is linear up to (0.5 - 0.6) g, and then increases in a nonlinear manner until reaching a maximum value. This value is given by the maximum lateral acceleration that can be present in a steady state cornering.

However, torque vectoring can extend the vehicle linear response zone, increasing the maximum lateral acceleration and modifying the understeer gradient (Grzegoek and

Weigel-Milleret, 2016). This is exactly what happens with the current NMPC implementation. In Figure 7.24 both an explanatory sketch (Subfigure 7.24a) and the modes comparison (Subfigure 7.24b) are shown.



(a) SWA - a_y (Grzegoek and Weigel-Milleret, 2016)



(b) Ramp steer test

Figure 7.24: SWA vs lateral acceleration

It can be seen in Subfigure 7.24b that a higher lateral acceleration is indeed obtained for the NMPC approach. Moreover, the linear zone is extended from 0.8 g to approximately 0.92 g. This result shows that for a given SWA the lateral acceleration will be higher leading to an improved behavior when cornering.

Chapter 8

Results

In this chapter, a set of open and closed loop maneuvers will be performed to test the NMPC behaviour under different maneuvers. Moreover, several simulations will be carried out in a real machine simulator to extend the validation of the NMPC.

8.1 Open loop maneuvers

In order to extend the NMPC validation carried out in Chapter 7, two open loop steering maneuvers will be tested.

8.1.1 Step steer test

The wheels torque behaviour can be seen in Figure 8.1. The sum of all torques is below the torque request signal throughout the whole maneuver. Moreover, the torque signals are within the allowed bounds. In this test, the inner wheels increase their torques when the step of the steering angle is produced.

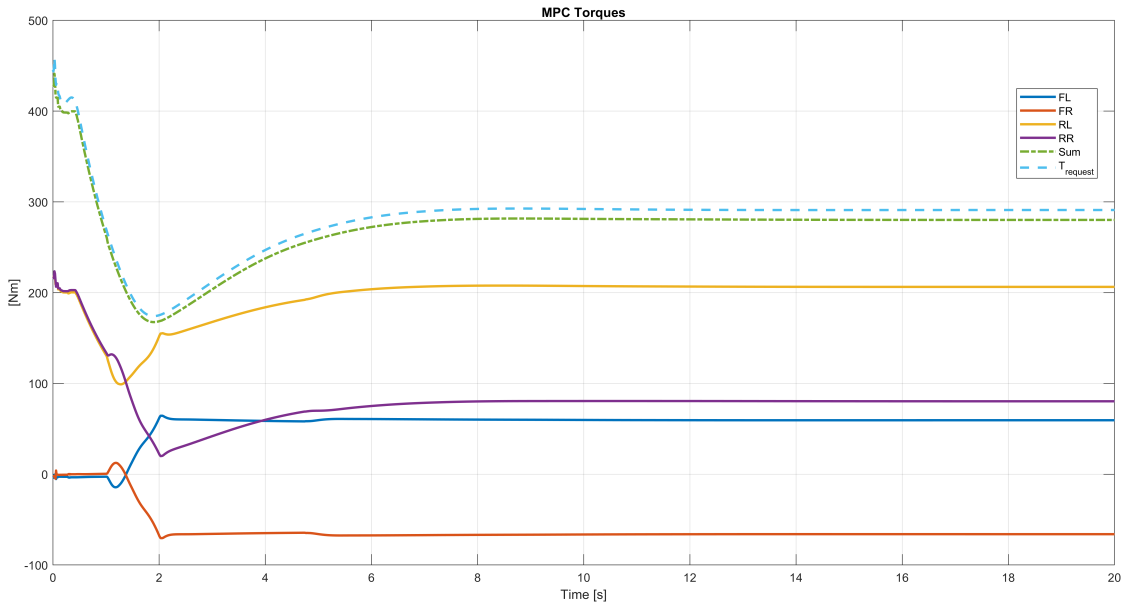


Figure 8.1: Wheels torque (step steer)

On the other hand, the torque rates and the states are shown in Figure 8.2. The torque rates are within the bounds and do not present an undesired behaviour but a smooth shape. The sideslip angle presents a very similar behaviour to that of the CRT plant. Then, both yaw rate signals (CRT and Simulink plants) are slightly below the yaw rate reference values. Therefore, the performance of the optimization variables is satisfactory.

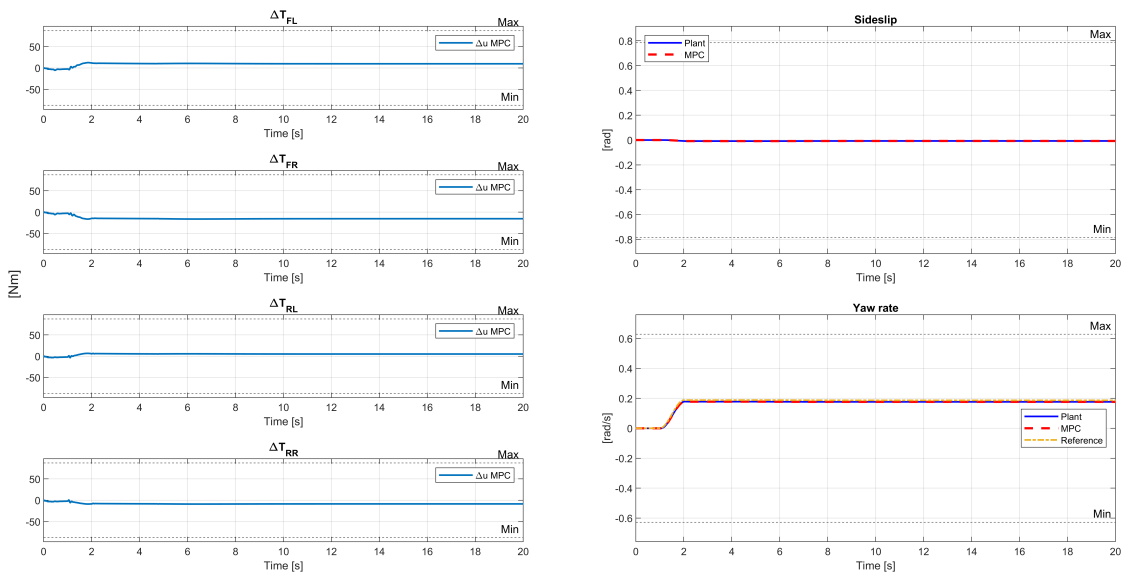
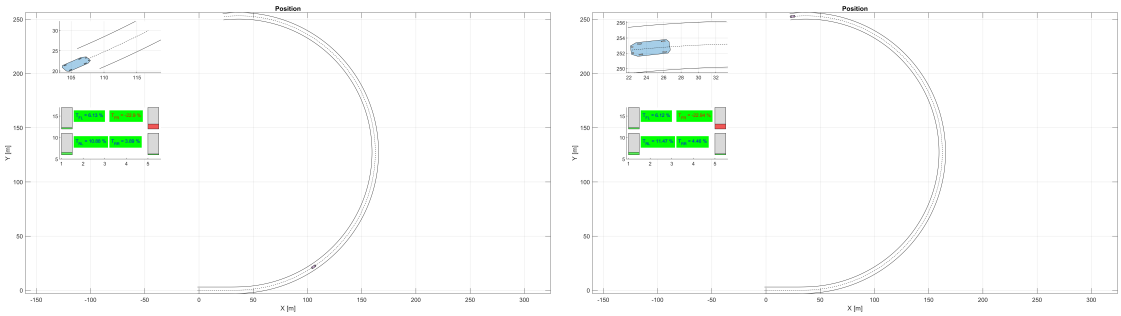


Figure 8.2: Torque rates and states (step steer)

Eventually, the step maneuver desired trajectory and current path travelled ares compared in Figure 8.3. In the subplots 8.3a and 8.3b, two steps of the maneuver are shown. In this maneuver, the error obtained for the car pose is lower than the one obtained for

the ramp steer maneuver. In fact, the car is almost perfectly centred with the path center line. This can be clearly seen in the upper left corner in Subfigure 8.3b. The effect of the steering angle also affects the turning radius and therefore the vehicle cornering as expected.



(a) Maneuver sequence 1/2 (b) Maneuver sequence 2/2

Figure 8.3: Step steer maneuver trajectory

8.1.2 Swept-Sine steer test

The next step is to test the swept-sine steer maneuver. The torque distribution can be seen in Figure 8.4. As the sinusoidal behavior prevents clearly seeing the evolution of each signal when they are all shown on the same graph, only a portion of the 20-second maneuver is shown. The sum of all torques is below the torque request during the maneuver and the torques are included within the bounds.

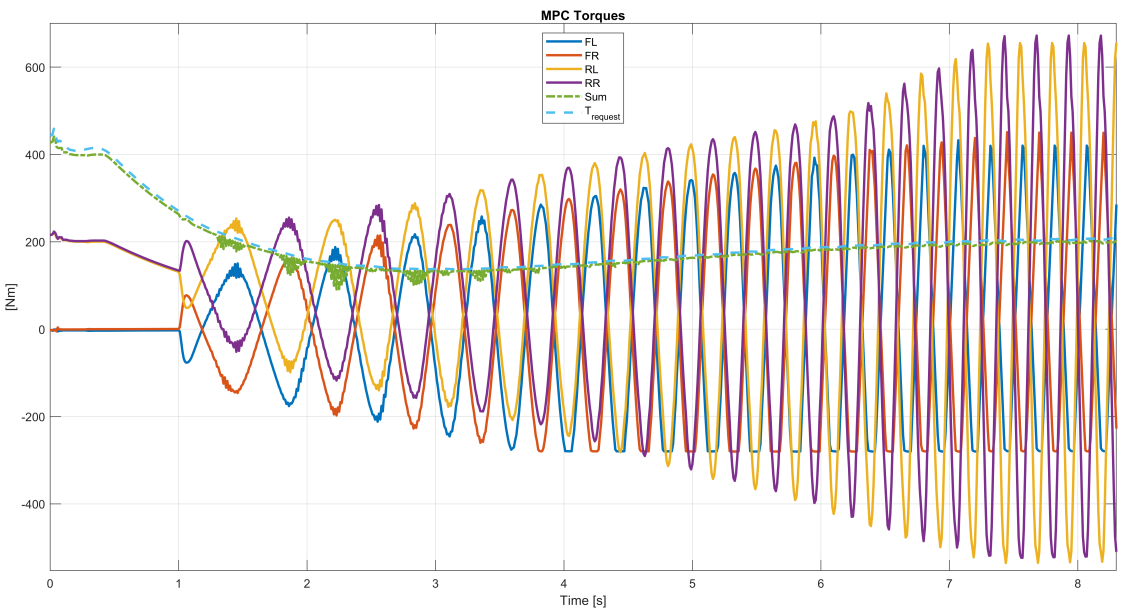


Figure 8.4: Wheels torque (Swept-Sine steer)

The variation of torques and the states are shown in Figure 8.4. The input rate of

each wheel is inside the bounds during the test. Regarding the states, the sideslip angle matches the CRT plant. Then, the yaw rate is slightly below the yaw rate reference as well as it occurs with the CRT plant value.

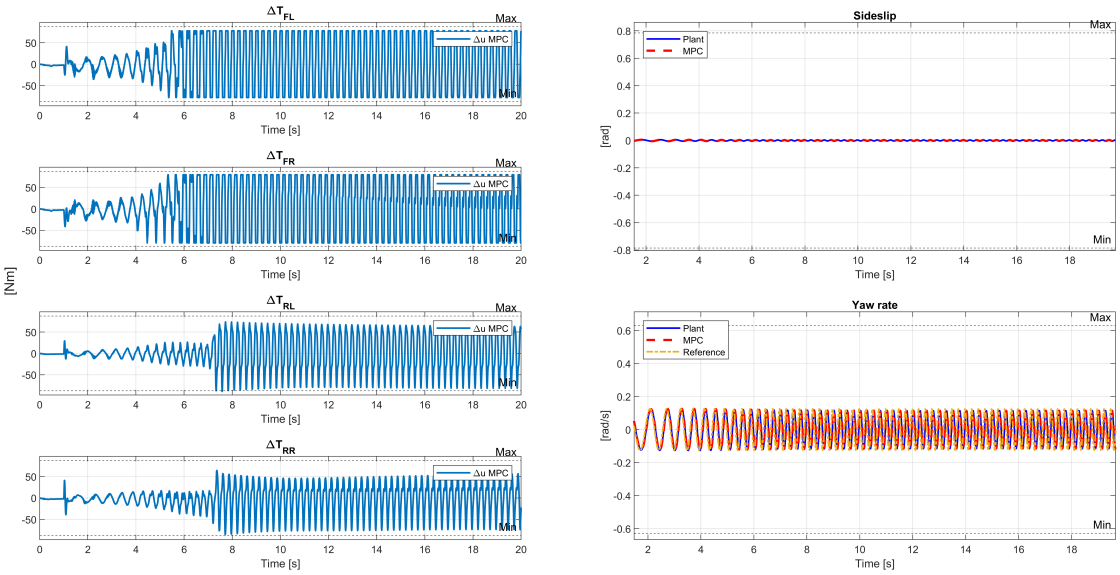


Figure 8.5: Torque rates and states (Swept-Sine steer)

The car trajectory plot shown for the open loop maneuvers is not shown since the torque varies sinusoidally and its value in each wheel would have to be analyzed every step according to the current phase of the signal.

8.2 Closed loop maneuvers

In this section, the NMPC validation through of the closed loop maneuvers is done. The Double-Lane Change (DLC) maneuver will be performed.

8.2.1 Double-Lane Change

The DLC maneuver is a standardized track test defined by the International Organization for Standardization (ISO) to provide repeatable and discriminatory test results while assessing the safety of a vehicle's response. The ISO 3888-1:2018 standard specifies the dimensions of the test track for a closed loop test method to subjectively determine a double lane-change which is one part of the vehicle dynamics and road-holding ability of passenger cars (ISO Copyright Office, 2016). The dimensions of the DLC track is shown in Figure 8.6. The lengths of track sections are fixed. The distance between cones of the same section depends on the vehicle width. The vehicle to be tested is driven through this track traveling a distance of 125 m.

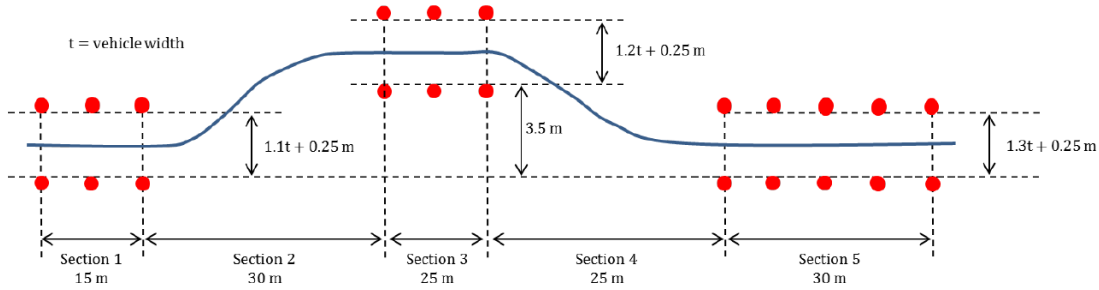
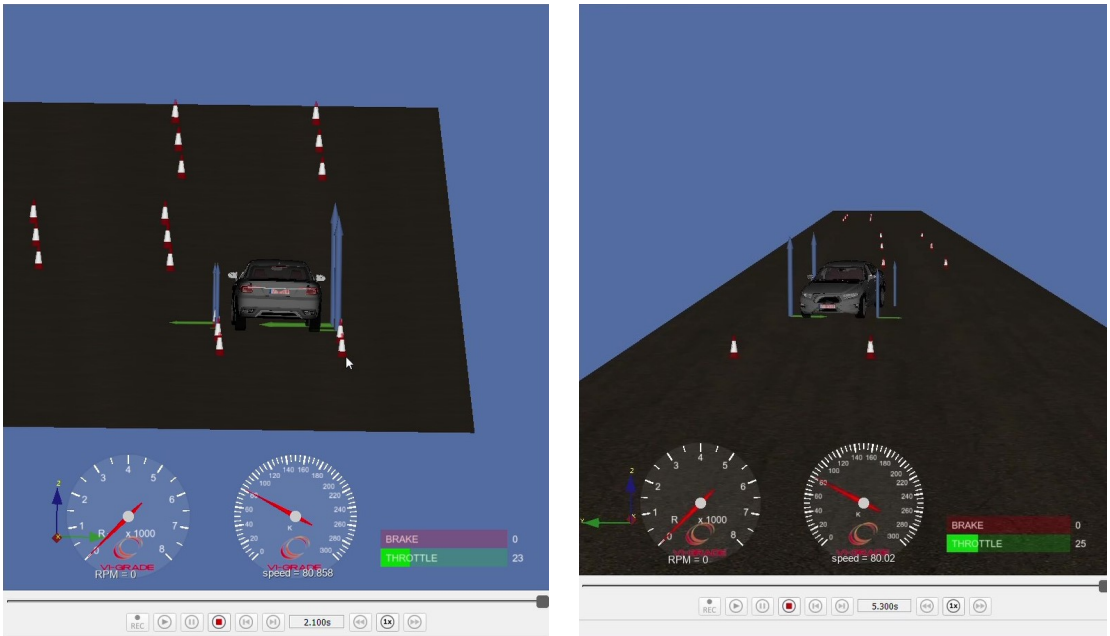


Figure 8.6: Double-Lane Change track dimensions

The DLC maneuver using the PressManeuvers event is shown in Figure 8.7.



(a) DLC sequence 1/2

(b) DLC sequence 2/2

Figure 8.7: DLC maneuver (VI-Animator)

The path travelled by the vehicle is shown in Figure 8.8. To facilitate the visualization of the plot, the cones and the limits of the lane have been added according to the ISO 3888-1:2018 standard. The vehicle is able to pass through a narrow passage successfully avoiding obstacles. Likewise, the vehicle has been added with the pose it had at different times during the maneuver.

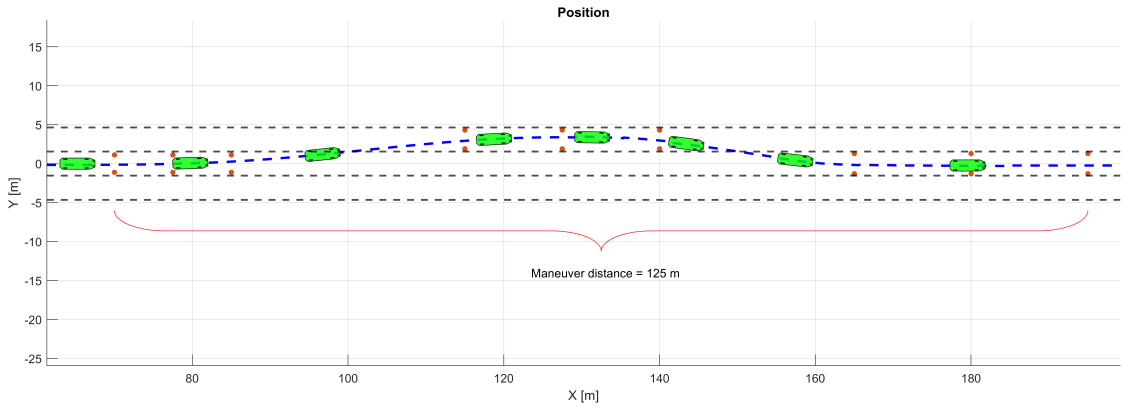


Figure 8.8: Path travelled (DLC maneuver)

In order to check if the solver is working properly, the information that is provided will be shown in Figure 8.9. The exitflag signal is kept at 1 (optimal solution found) throughout the maneuver.

Regarding the time elapsed until the solver finds an optimal solution, it must be said that is far below the sampling time limit, which ensures the real-time performance of the controller implementation.

The objective function cost does not experience a large change while the maneuver is performed. In fact, the values are around 10^7 and it may seem that a very large error accumulates, but this is not the case, since the error increases each time instant ($1\,ms$) and the control action is only applied every $10\,ms$).

The DLC test starts with a longitudinal velocity of $85.4\,km/h$ and it decreases slightly when passing through the second narrow passage and it increases afterwards as going through the last set of cones.

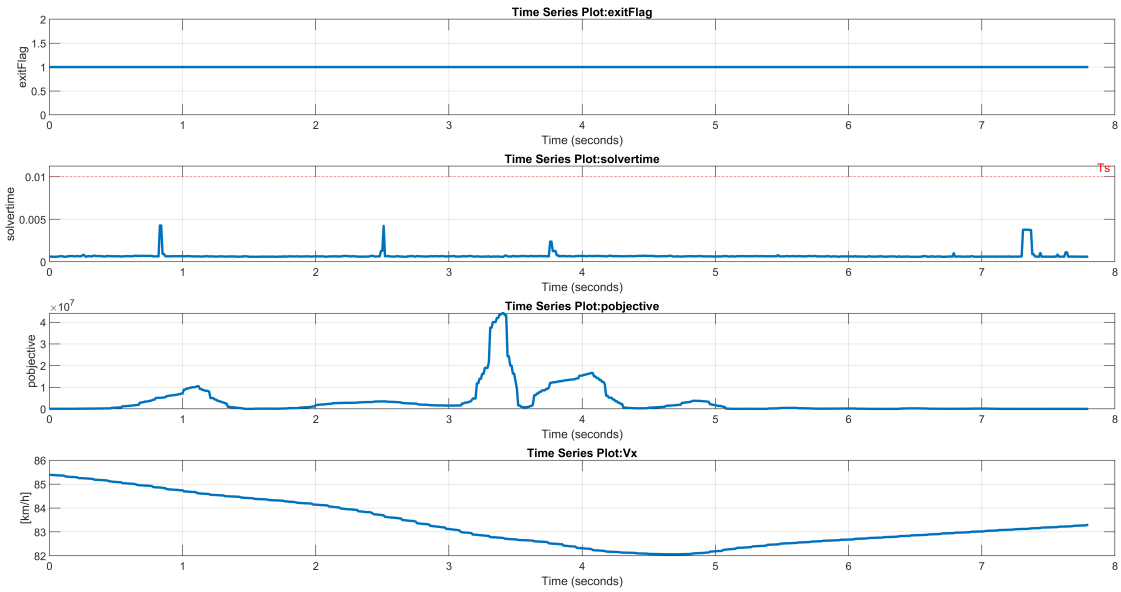


Figure 8.9: Solver information (DLC maneuver)

The wheel torque and torque rate evolution is shown in Figure 8.10. The wheel torque signals differ from the torque request in each wheel as expected, since the closed loop scheme (the plant is communicated with the maneuver optimizer) ensures taking into account the previous states to compute the new references which leads to improve cornering, and therefore, maximizes the use of torque on wheel motors. The input rate signals are within the bounds as well as the wheel torque signals and show a smooth behaviour over time.

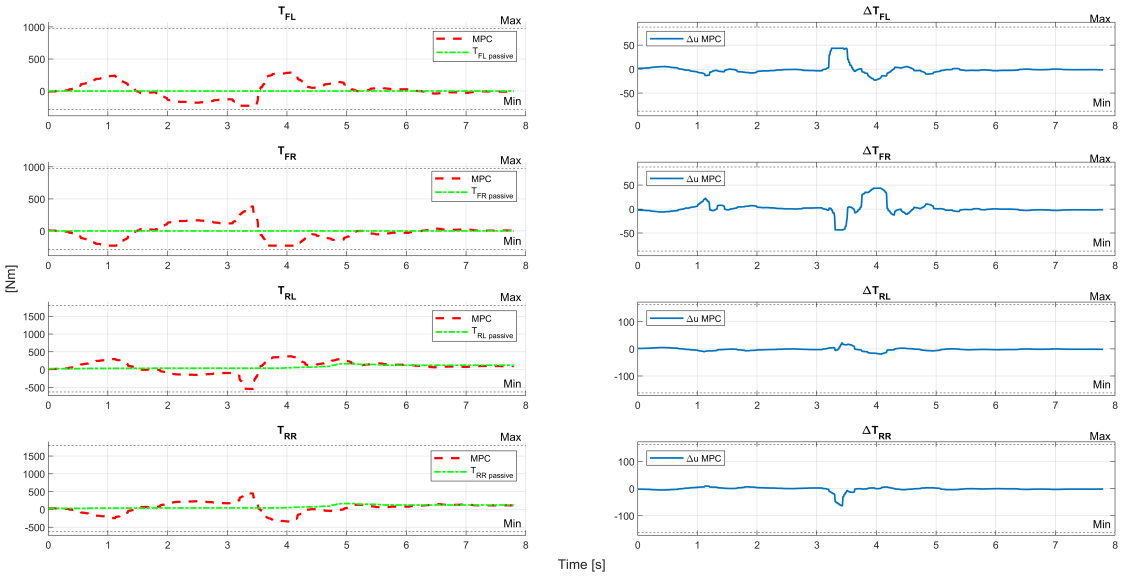


Figure 8.10: Wheel torque and torque rate (DLC maneuver)

The torque distribution is observed in more depth in Figure 8.11. The sum of torques is very close to the torque request over time. The left wheels apply more torque to compensate the negative SWA of the VI-Driver to ensure not loosing the vehicle control while going trough the first set of cones. Then, just the opposite happens to face the start of the second section of the maneuver where more torque is delivered to the right wheels. When exiting this narrow passage more torque is delivered to the outer (left) wheels to have the vehicle positioned to pass the last set of cones.

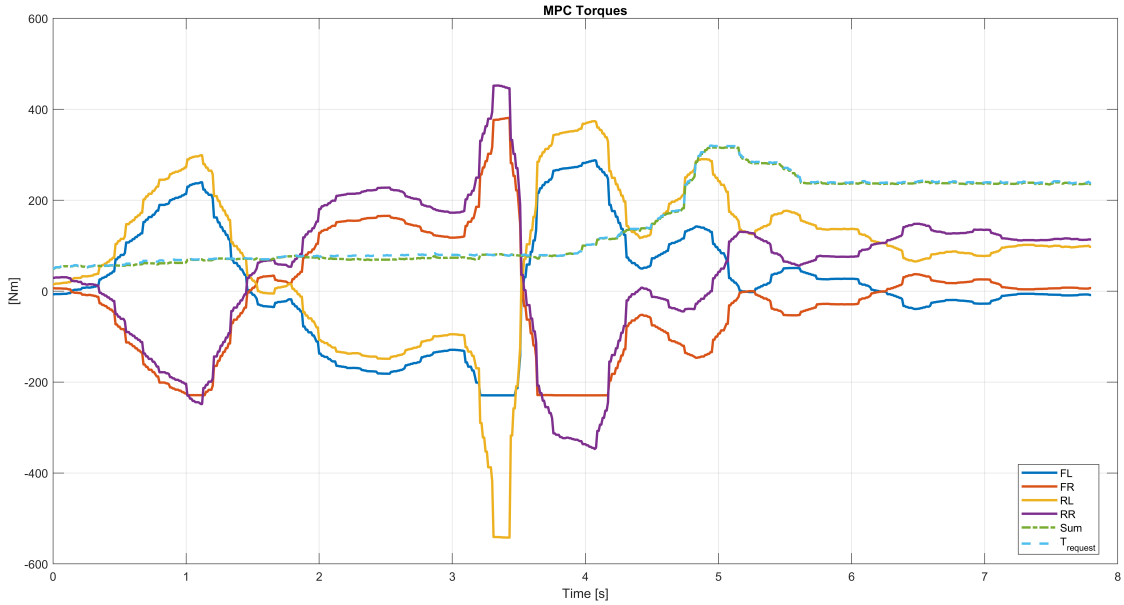


Figure 8.11: Wheel torque distribution (DLC maneuver)

Regarding the yaw rate, its evolution can be seen in Figure 8.12, in which the optimized yaw rate is very similar to the reference value, being slightly lower as it has happened in previous maneuvers. The yaw rate is within limits at all times. Therefore, the execution of the closed-loop maneuver has been satisfactory.

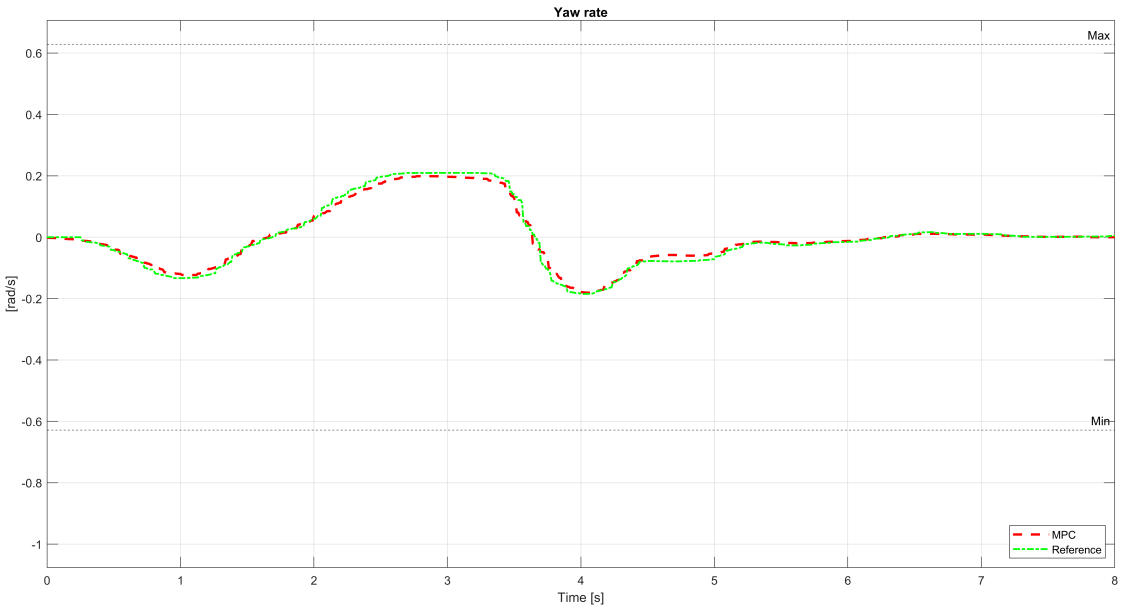


Figure 8.12: Yaw rate (DLC maneuver)

8.3 Driving simulator

Once the offline tests have been carried out, the integration of the NMPC implementation in the real-time machine simulator (VI-Grade Compact Simulator) will be carried out. The aim is to test the vehicle performance through various dynamic driving conditions.

Two different scenarios will be analyzed:

- Free driving and constant radius circle turn on the Proving Ground track.
- Perform a sequence of DLC maneuvers (one after another) in a three-lane track.

The driving simulator and a loaded scenario are shown in Figure 8.13. But before delving into testing the vehicle in the scenarios, it is necessary to highlight the introduction of certain key aspects that have improved the current implementation.

(a) Driving simulator

(b) Driving scenario loaded

Figure 8.13: Driving simulator testing

In Chapter 7, it was mentioned that the `TV_activation` flag was used to select both the passive mode (without controller) or the TV mode when performing a maneuver. However, it does not make sense to use the TV mode for the entire velocity range of the vehicle but for a certain range either performing tests in a real scenario or in a driving scenario. In the latter, the inputs are not given from the execution of the maneuvers (driver inputs are parameterized in advance) but by a person in a driving simulator.

When switching between modes, there should be no sudden change between them which would result in an undesired behaviour (noticeable offsets) due to the dynamic difference the vehicle undergoes when driving passively or controlled.

In order to smooth the mode switching, a LookUp table considering the longitudinal speed and the the torque error weight, Q_{e_u} has been designed. The passive mode is active for low speeds ($v_x < 30\,km/h$) and there exists a gradual progression from the passive mode to the TV mode ($30 \leq v_x \leq 40\,km/h$). In this range, Q_{e_u} decreases progressively from the passive mode set value to the TV mode one, assuring a soft transition between modes, when velocity increases and when it decreases. So, the TV mode is applied when ($v_x > 40\,km/h$).

By default, the vehicle operates using this logic according to the longitudinal velocity but the user can always switch manually between modes through the `TV_activation` flag when wanted.

On the other hand, in order to visualize the system variables online, and also to the simulation data, a Simulink scheme is running in co-simulation with the driving simulator. The communication of signals between both systems is carried out through the UDP protocol (User Datagram Protocol).

8.3.1 Scenario 1

Testing in the first scenario is mainly aimed to perform free driving maneuvers and make turns over a constant radius for both driving modes, to see if the controller helps to drive when cornering and a higher velocity is achieved when cornering without losing the vehicle stability. The path travelled is shown in Figure 8.14. The trajectory described by the vehicle corresponds to the dashed lines. In order to have a clearer view of the path followed, two continuous lines as track limits have been plot on the vehicle sides.

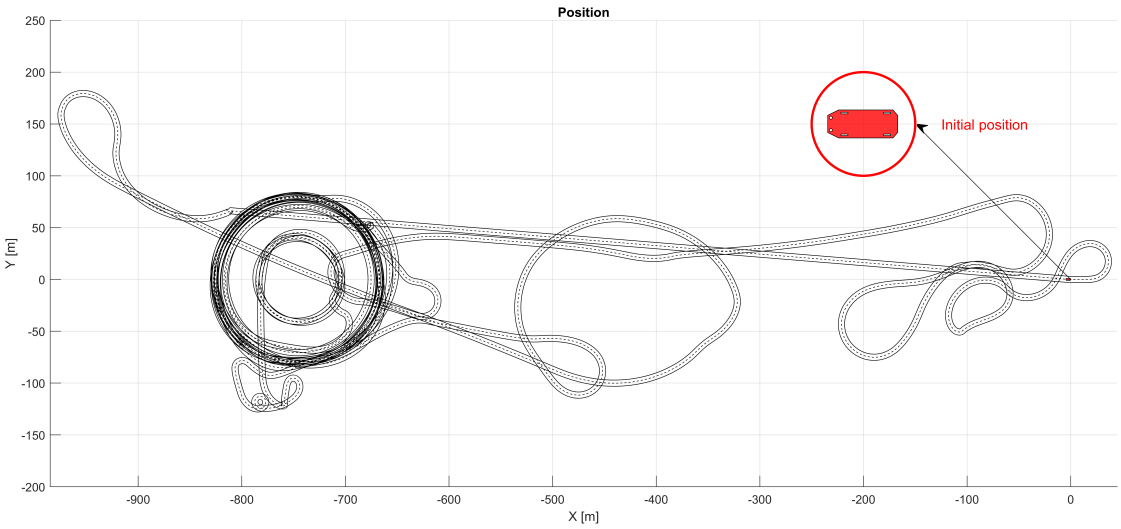


Figure 8.14: Path travelled in the track (scenario 1)

The test starts by doing free driving on a section of the test track to test the vehicle dynamic response when switching between modes. Subsequently, the passive mode is tested by making turns with a constant radius and increasing the speed until, by moving the steering wheel, it is not possible to maintain the circular path.

In Figure 8.15, some key aspects of the solver data are shown. The exitflag signal is kept at 1 over time except in those moments in which the switching is produced from the TV mode to the passive one since the TV control module is disconnected.

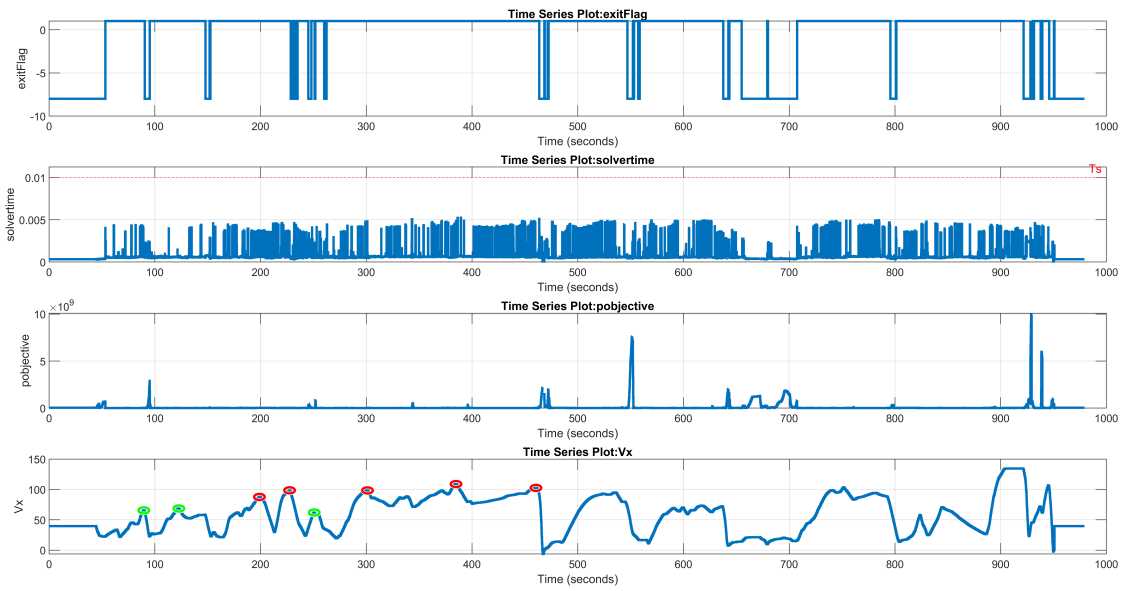


Figure 8.15: Solver information (scenario 1)

It can be seen that the time needed to solve the problem each time instant is below the controller sampling time leading to a computing time that is quicker than the real-

time response constraint. The objective function values are also shown, appearing error peaks when the change of modes occurs due to the increase of the torque error weight (it increases two orders of magnitude).

Moreover, the longitudinal velocity is shown in the last subplot. The highlighted signal peaks are the maximum velocities achieved while maintaining the constant radius path. The green ones correspond to the use of the passive mode and the red ones to the TV mode. A higher velocity is achieved for the latter mode, starting from almost 70 km/h in the first mode to almost reaching 110 km/h in the second one. Therefore, TV mode allows cornering at higher speeds without losing vehicle stability.

In order to illustrate better the vehicle behaviour when switching between modes, Figure 8.16 is shown. In here, after braking, the velocity is under the limit, so the passive mode is activated. When the longitudinal velocity is reduced below 30 km/h, the TV mode is disconnected which could lead to an exitflag signal of -8 (optimal solution not found) since the tire model is not accurate for low speeds. A very accurate tire model requires a specific controller to manage the tire dynamics, which would greatly increase the computational demand and is outside the scope of this thesis.

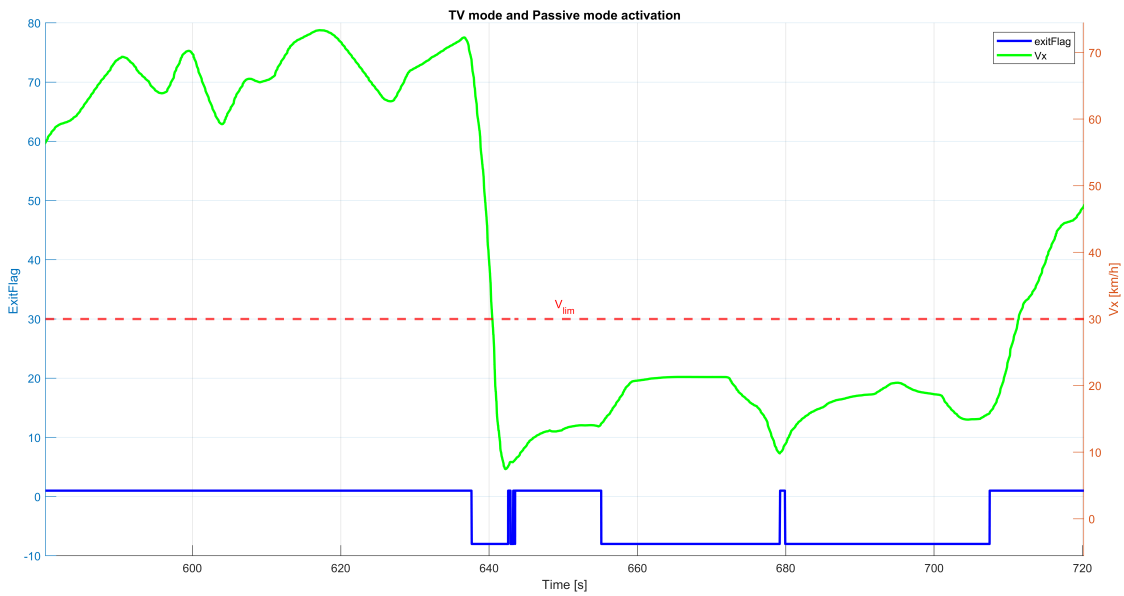


Figure 8.16: Driving mode activation (scenario 1)

The evolution of wheel torque and the torque rate is shown in Figure 8.17. The optimized wheel torque does not match the passive torque reference especially on the front wheels which means that TV is being applied. To see if the wheel torque ease the driving, the wheel torque will be analyzed deeply afterwards. As for the torque rate, the bounds are not reached nor undesired changes between time instants are observed despite the duration of the test, which is very positive to carry out a smooth control.

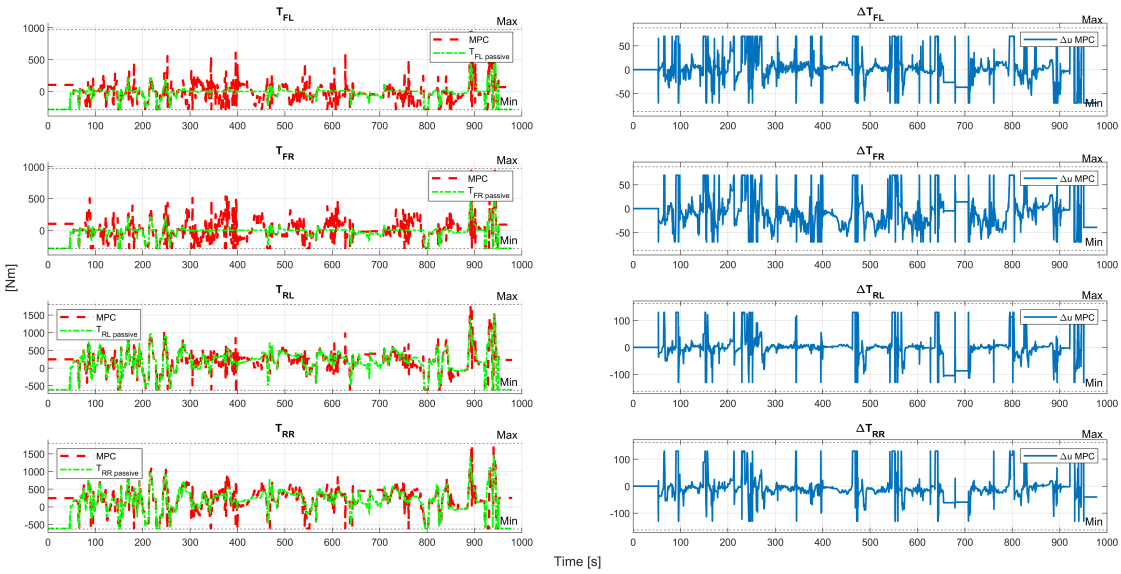


Figure 8.17: Wheel torque and torque rate (scenario 1)

In Figure 8.18 it is shown a time interval of the test, in which the vehicle is performing a circular path at high speed while the TV mode is active. The wheel torque distribution is depicted and it can be seen that the sum of all wheel torque is always below and close to the torque request signal ensuring the requested torque delivery. Moreover, the distribution of torques is smooth and a higher torque on the right wheels at $t_1 = 430\,s$ and on the left wheels at $t_2 = 490\,s$ is leveraged to facilitate driving in left and right turns, respectively.

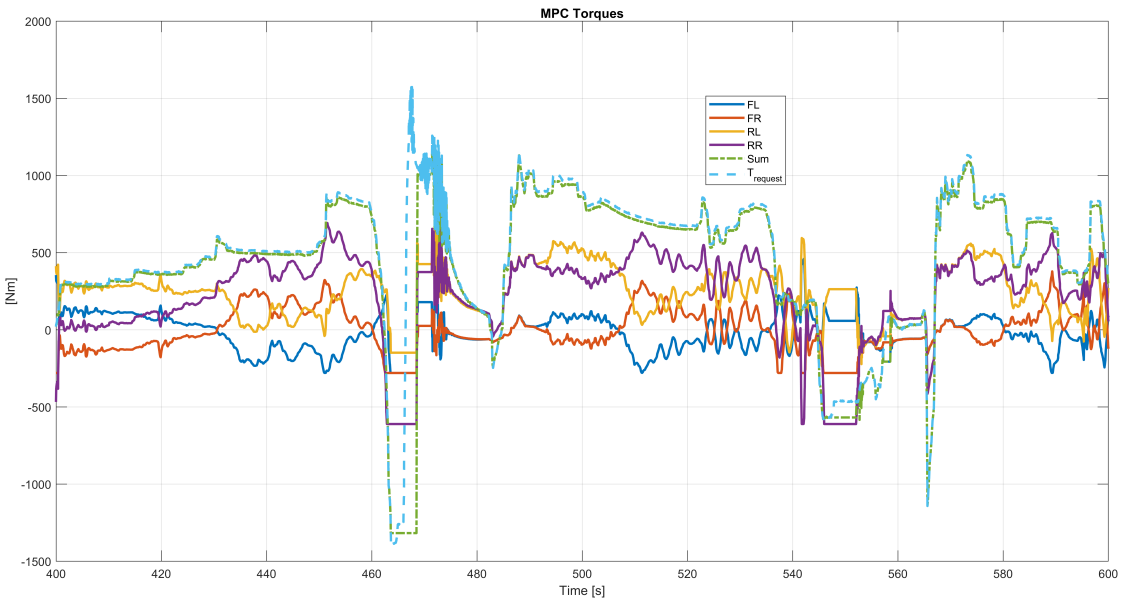


Figure 8.18: Wheel torque distribution (scenario 1)

Eventually, the yaw rate evolution will be shown in Figure 8.19. The yaw rate is

slightly below the yaw rate reference in some time intervals and matches the yaw rate reference the rest of time, as it happened in the offline simulations before and after the yaw rate got saturated. This result can be clearly seen in the zoomed box of the plot just after $t = 580\,s$. Thus, at certain times the vehicle will not turn as much as the reference value, so there will be a small error in the path traveled. In addition, the optimized values are within the bounds throughout the test.

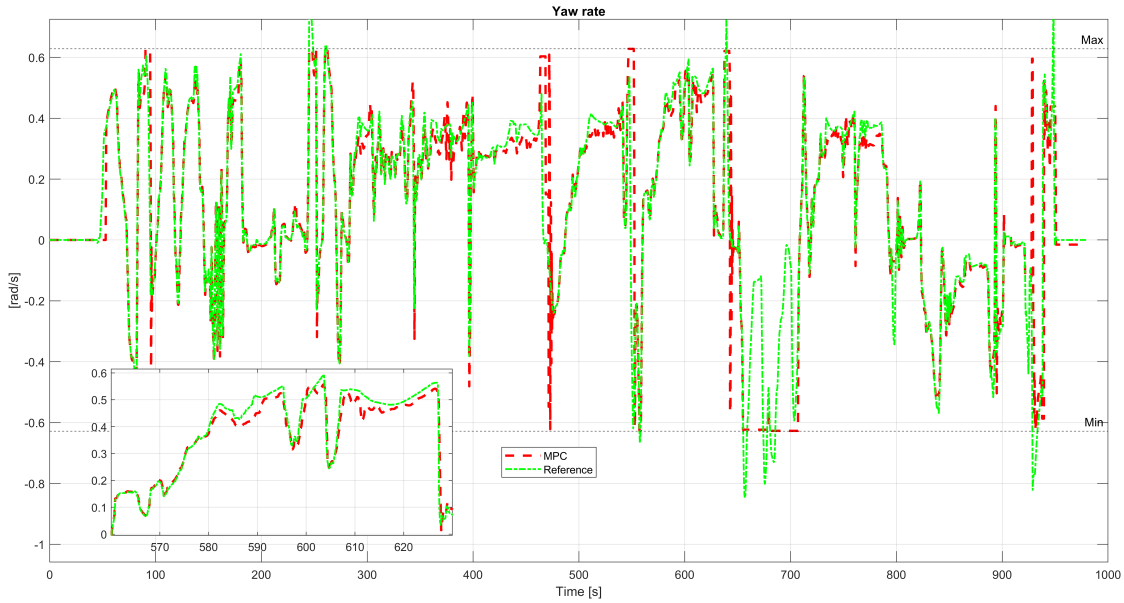


Figure 8.19: Yaw rate (scenario 1)

8.3.2 Scenario 2

Moving onto the second scenario, the aim is to perform DLC maneuvers (to the left and to the right) just one after the other in a three-lane track. The approach is to continually increase speed until the maneuver cannot be accomplished without avoiding obstacles, which in this case are traffic cones. In this test, only TV mode will be tested as passive mode has already been tested in the first scenario. The path travelled is shown in Figure 8.20. The trajectory described by the vehicle corresponds to the continuous line and lane boundaries are shown with dashed lines.

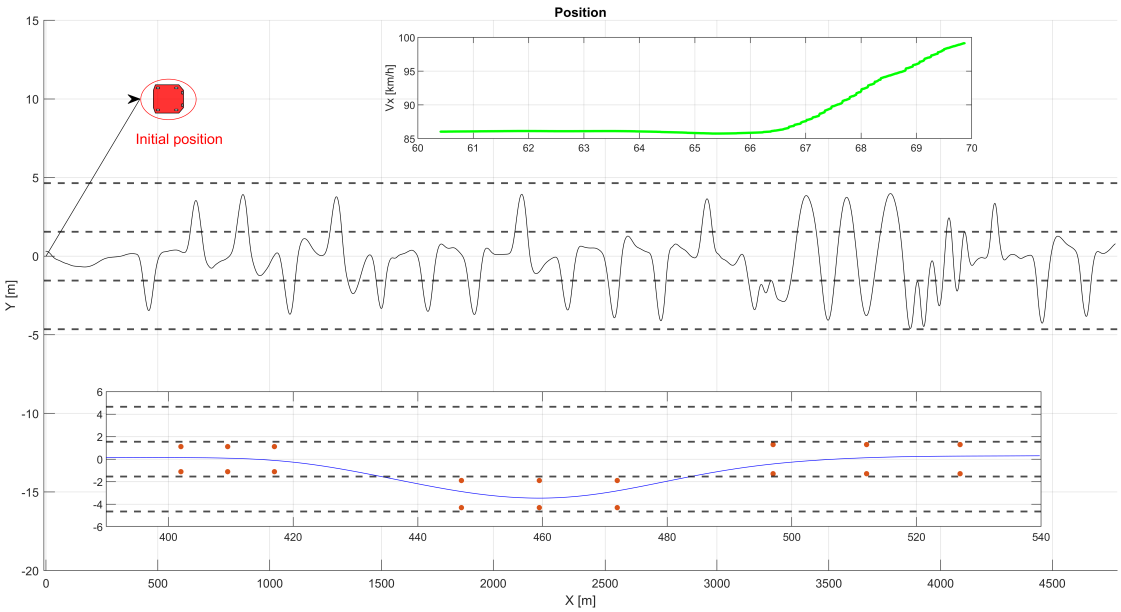


Figure 8.20: Path travelled in the track (scenario 2)

The starting longitudinal velocity is 80 km/h and the fastest DLC maneuver without losing the vehicle control was performed at a velocity of 104 km/h. Moreover, one of the DLC maneuvers is depicted in a zoomed box showing the cones that are placed following the directives provided by the ISO 3888-1:2018 standard. The vehicle velocity performing this maneuver can be seen at the top of the plot.

The information provided by the solver is shown in Figure 8.15, some key aspects of the solver data are shown. The exitflag signal is kept at 1 throughout the whole test except before $t = 36\,s$ which is just before starting the test (vehicle was driven to reach the start of the scenario track).

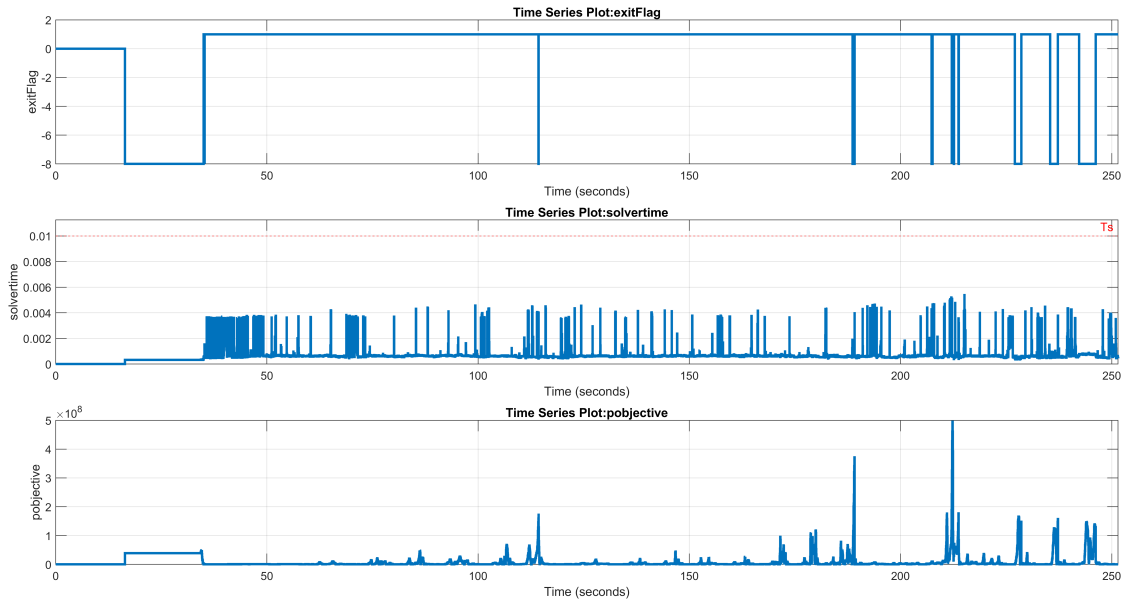


Figure 8.21: Solver information (scenario 2)

In addition, the time needed to solve the problem each time instant is always below the controller sampling which ensure the real-time performance. Then, the objective function values are also shown, appearing error peaks an optimal solution was not found due to the fact that vehicle reached driving limits. In this case, the error peaks result in an increase of one order of magnitude, not two, since the passive mode is not activated.

The evolution of wheel torque and the torque rate are shown in Figure 8.22. As expected, the optimized wheel torque does not match the passive torque reference over time, especially on the front wheels where the torque allowed delivery is lower than the rear wheels since the rear axle motors can provide more torque (65 % of the total torque) and the torque distribution proportionality constants introduced in the objective function by means of the torque rate weight, Q_{r_u}. Regarding the torque rate, it must be said that the bounds are not reached nor undesired changes between time instants are observed.

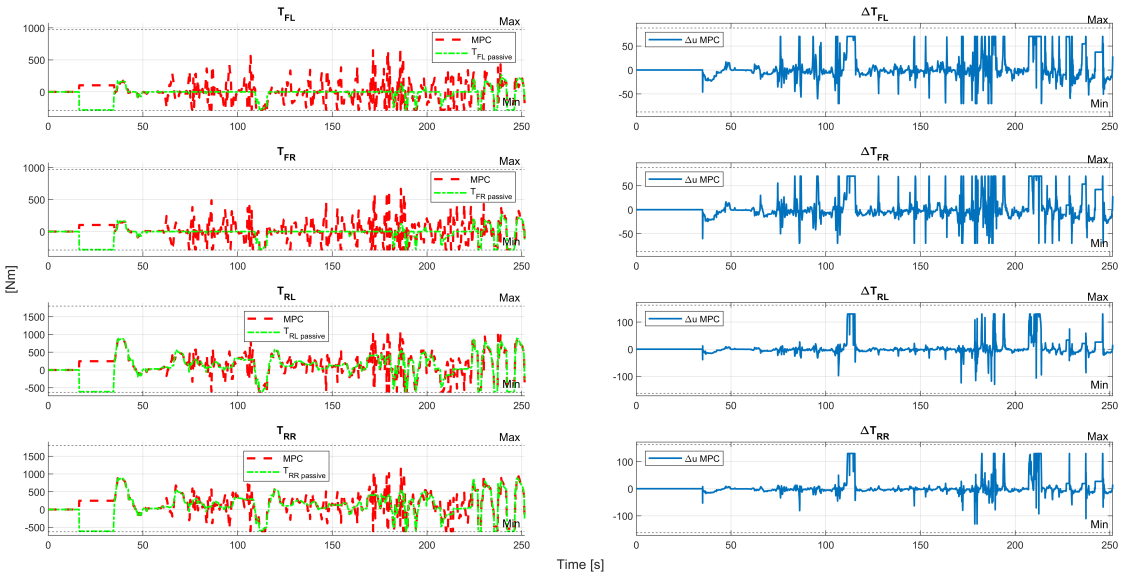


Figure 8.22: Wheel torque and torque rate (scenario 2)

In Figure 8.23 it is shown a time interval of the test, in which the vehicle is performing a DLC maneuver. The wheel torque distribution is depicted and it can be seen that the sum of all wheel torque is always below and very close to the torque request signal ensuring the requested torque delivery. Moreover, the distribution of torques is smooth. A higher torque is applied to the right wheels at $t_1 = 62.5\,s$ and to the left wheels at $t_2 = 65.5\,s$ to compensate the SWA given by the driver, respectively. The distributed torque delivery enables the vehicle to be stable while passing through a narrow passage at high velocity and also, avoid oversteering which would lead to the vehicle loss of control.

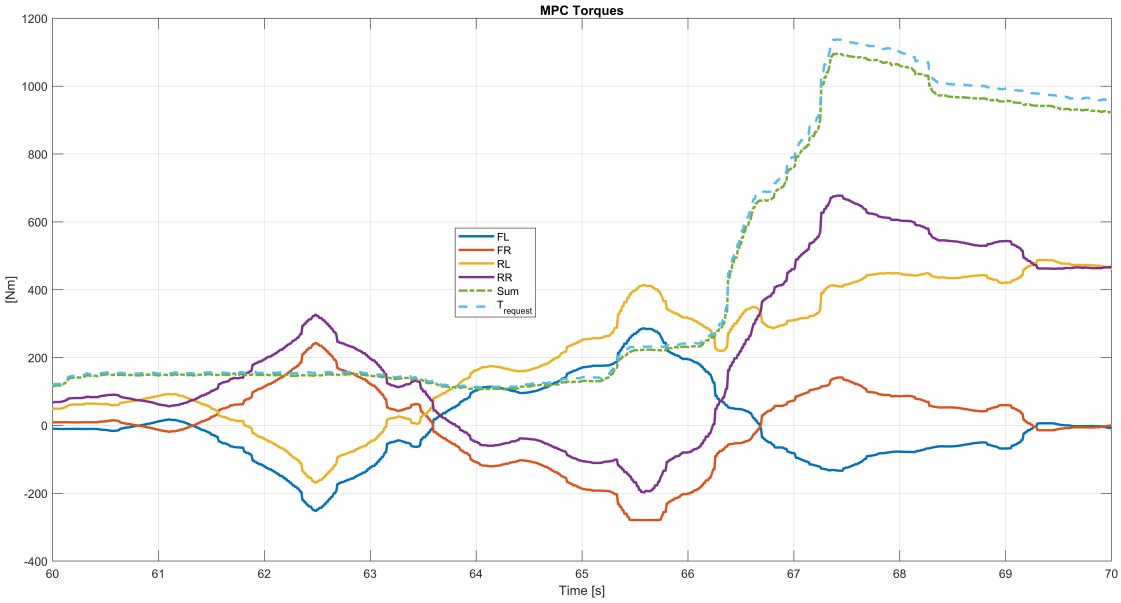


Figure 8.23: Wheel torque distribution (scenario 2)

The last system variable analyzed is the yaw rate which is shown in Figure 8.24. The yaw rate is slightly below the yaw rate reference in some time intervals and matches the yaw rate reference the rest of time, as it happened in the first scenario and while performing the offline simulations. The yaw rate reference and the optimized yaw rate can be clearly seen in the zoomed box of the plot just after $t = 80\,s$. Additionally, the yaw rate values are within the bounds throughout the test duration.

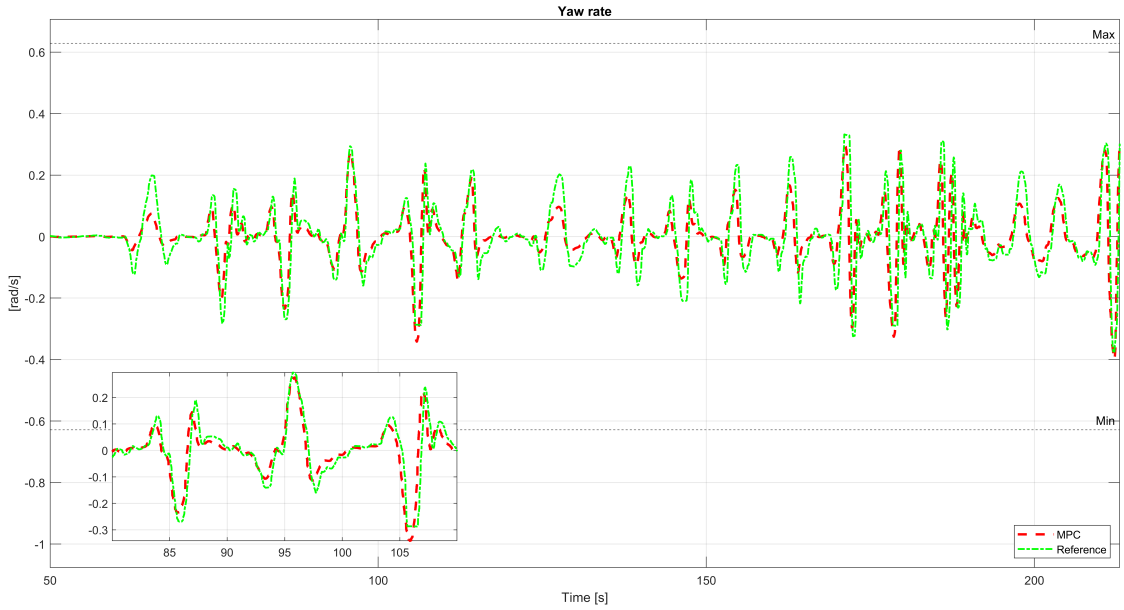


Figure 8.24: Yaw rate (scenario 2)

8.4 Outcomes

The TV-NMPC has been successfully validated both through offline simulations and using the driving simulator. The yaw rate closely follows the reference value, being slightly below and above it when the signal is positive and negative, respectively. The optimized sideslip angle matches almost 100% the plant (vehicle) signal. When the optimized sideslip angle enters the plant, the output sideslip coincides at a great extent.

Regarding the torque distribution, it was as expected, leading to a smooth behaviour that allows to improve vehicle handling and getting higher velocities when cornering without losing the vehicle control.

Eventually, it has been confirmed that the implemented system is capable of performing the computations faster than the sampling time of the controller, therefore, the viability of the real-time specification of the controller has been validated.

Chapter 9

Environmental, economic and social impact

In this chapter, the project sustainability will be addressed, with special emphasis on the environmental, economic and social impact.

9.1 Economic impact

EV provide economic benefits by reducing fuel costs and shifting consumption away from imported oil to less contaminant electricity sources. EV users can expect to save between 950 € and 1250 € annually on fuel costs, totaling between 11400 € and 15000 € over the life of the vehicle (12 years), depending on the fuel price (SWEEP, 2014).

There is no doubt that electric vehicles and their direct relationship with autonomous vehicles are having a huge impact in the way the economy works taking part in the new industry 4.0. In the last decade, vehicle manufacturing companies have developed efficient procedures and installed robotic systems to increase productivity, reduce the running time and cutting down on costs and expenses, which leads to an economic growth increment.

In addition, modern and intelligent control techniques also make it possible to work in a wide variety of operating conditions which translates into shorter cycle times and cost savings. A study (McKinsey, 2013) shows the potential economic impact of different disruptive technologies, with the autonomous vehicle and advanced robotics as two of the technologies that have a higher impact on global economy.

9.2 Social impact

Electric cars are already entering next disruptive market force for both transportation and technology. One of the key aspects is that electric cars do not emit as many harmful gases as gas vehicles, and thus do not emit chemicals into the air. Societal benefits for EVs include better air quality and health, reduced healthcare costs, job creation benefits, and environmental benefits (Malmgren, 2016).

Electric cars includes mainly a battery and electric engines, which reduces the amount of mechanical parts that gas vehicle have and, therefore, the chance of breaking is decreased. This carries out a maintenance cost reduction.

Technology growth goes hand in hand with the vision of society in their progression as a community and tastes. The recent advances in control engineering have had an impact on the design of complex systems with highlighted safety measures and error handling strategies, which can prevent both people from having accidents and damages to property.

Society is living in a transportation era, where cleaner and also affordable vehicles are a reality. Embracing this EV revolution will not only help environmental issue, but also benefits public health, transportation efficiency and fuels technological innovation.

9.3 Environmental impact

The consequences of a widespread use of electric vehicle are significant. As all technologies involving innovation, microprocessors and therefore, semiconductors, are essential components for its functioning, and the manufacturing of these elements is highly polluting. Energy saved from optimal torque distribution could decrease to a great extent the emissions by reducing battery energy consumption and making efficient driving. Advanced control strategies could have a significant result in reducing the environmental impact and polluting emissions (Eom et al., 2006).

For the analysis of the environmental impact of the project, the laptop used for its development is taken into account. The VI-Grade Compact Simulator environmental impact is not considered since there is no consumption data available. For a rigorous analysis, each component and device (monitor, lights, ...) consumption and emission must be considered but the energy consumption tracking is not plausible, so only the laptop will be considered for this analysis.

According to data from the European Commission, a computer emits between 52 and 234 equivalent grams of CO_2 considering a power of between 80 and 360 Watts every hour that it is turned on, according to the European Commission. Although this includes all types of computers, the French Environment Agency ADEME (ADEME, 2018) has estimated that a laptop consumes between 50 and 80 % less than a desktop computer.

Taking as a reference value the average emission of a computer and the average of the estimation of the percentage consumption of a laptop, an emission of 93 g CO_2/h is obtained. Thus, the total carbon footprint value of the laptop used in the project is given by

$$CO_2 \text{ emissions} = 93\,\frac{g\,CO_2}{h} \cdot \frac{1\,kg}{10^3\,g} \cdot 1040\,h = 96.72\,kg\,CO_2 \tag{9.1}$$

Chapter 10

Budget

In order to estimate the budget to finance this thesis, it is needed to take into account all the material and human costs invested in the project, these are organized into three groups: costs related to licenses, costs related to the work carried out (hours spent for the development) and costs related to the amortization of the equipment used. The costs will be computed estimating a total working time 1040 hours within the 8 months period devoted to develop the project. The engineer wage has been estimated to 15 €/h.

In addition, some of the license costs are confidential, so the approximate cost values will be considered. Likewise, the cost of the real machine simulator is not available but its concept must be included in the cost structure that is shown in the Table 10.1.

Concept	**Unit**	**Cost (€)**
Matlab license (including toolboxes)	1	2500
Simulink license	1	1200
ForcesPro license	1	2000
VI-CarRealTime license	1	3500
VI-Grade Compact Simulator	1	-
GitLab license	1	100
Engineer wage	1	15600
Amortization (laptop)	1	213.33
Total cost (€)		25113.33

Table 10.1: Project cost structure

Moreover, the amortization cost of the laptop used in the development of the project was taken into account. Bearing in mind the Law 27/2014, of November 27, on Corporation Tax. (BOE, 2022), the maximum annual percentage of amortization for electronic equipment is 20 % and the maximum amortization period is 10 years. Moreover, the computer used has an approximate cost of 1600 €, leading to a total amortization cost

during the the duration of the project (r_p, ratio of project duration per year) of 213.33 € (see Equation (10.1)), which would match with the value which is obtained applying the maximum annual percentage of amortization if the duration of the project were one year (320 €).

$$\text{Amortization cost} = \frac{\text{Acquisition price}}{\text{Useful life}} \cdot r_p = \frac{1600\,€}{5\,\text{years}} \cdot \frac{8}{12} = 213.33\,€ \tag{10.1}$$

The project was started from scratch and the approximate budget invested (not all devices used can be quantified) to reach an implementable solution has been 25113.33 €.

Chapter 11

Conclusions and future work

This thesis has proposed some contributions to the vehicle modeling, parameter estimation, model predictive control and reference generation areas of the electric vehicle driving field, paying particular attention to both model and controller validation leveraging offline simulations and a real-time driving simulator. This chapter summarizes the work presented in this thesis in order to review the main conclusions and explore the possibilities of further research.

11.1 Conclusions

Estimation, plant modeling and control strategies for electric vehicles have been a main research field throughout the last years, and several theoretical and experimental results have been discussed in the literature. However, there are still many aspects to investigate and others to refine to improve vehicle stability and comfort, as well as to integrate more active subsystems that work in a coordinated manner through a generalized control layer. This thesis has contributed to the advance of the state of the art of this field.

- **Chapter** 3 has presented the general ICC control framework to address the context of the problem at a high level, with an special emphasis to the elements that intervene in the system and how the interaction with each other is. Then, it has been stated the main differences between the ICC structure and the TV presented structure and the considerations taken into account to improve the vehicle behavior. Finally, the controller specifications have been detailed.
- **Chapter** 4 has addressed the variety of dynamic models considered that encapsulate most of the motion dynamics present in a real vehicle from a non-linear perspective. The simulation plant as well as the real vehicle model have been

presented. Then, the vehicle stability and handling is analyzed to assess the specific behaviour of the vehicle. Eventually, the reference generation is described by means of the yaw rate reference approaches and the torque request through a pedal (torque) map.

- **Chapter** 5 has shown the techniques and driving tests to obtain the vehicle model parameters. For this purpose, physical measurements and the estimation of parameters have been carried out. Both linear and nonlinear least-squares approaches have been leveraged to identify vehicle parameters that are involved in the vehicle dynamics formulations and in the generation of references.
- **Chapter** 6 has proposed a real-time TV-NMPC approach for controlling an electric vehicle, focused on maximizing torque usage on wheel motors while improving handling capabilities. Then, the objective function terms and constraints are highly detailed to assess their significance. Afterwards, the different solvers used are compared and their particular formulations are developed. Eventually, the simulation and driving simulator implementation capabilities of the solvers is presented.
- In **Chapter** 7, the implementation details and systems are described as well as delving into the considerations taken into account to develop the subsystems interactions. The design of the offline open and close loop driving maneuvers is described. Later, the model variables are validated by means of several driving maneuvers (step steer, swept-sine, etc). The MPC tuning is detailed as well as a preliminar validation through a ramp steer maneuver. Finally, the NMPC approach is compared to the passive mode in which the controller is not active.
- **Chapter** 8 has addressed a set of open and closed-loop maneuvers to test the NMPC performance under different maneuvers. Moreover, several simulations in different scenarios will be carried out in a real machine driving simulator to extend the validation of the NMPC. The driving simulator is essential to test the vehicle behaviour while performing a DLC maneuver and make turns over a constant radius for both driving modes (passive and TV).
- **Chapter** 9 provides an overview of the project sustainability, with an emphasis in the developed systems impact in the environment, economy and society. Moreover, the total carbon footprint is taken into account by means of the CO_2 emissions.
- In **Chapter** 10, the budget destined to finance this thesis is estimated. All the material and human costs invested in the project, these are organized into three groups: costs related to licenses, costs related to the work carried out and costs related to the amortization of the equipment used.

11.2 Future work

This section resumes the open issues that could be addressed and/or upgraded in future work.

- Incorporate new control systems (ESC, ABS, TCS) to the ICC control structure by using MPC or control logic suitable for active systems with the aim of managing the control systems not individually but jointly. This aims to merge active systems encouraging the transition from usual electric vehicles to full autonomous vehicles.
- Address the low velocity performance of the vehicle by means of the design of a specific controller to fulfill this task.
- Generate useful references to further refine model fidelity, such as the sideslip reference to enable the vehicle sideslip angle and its rate to be within the stable phase plane zone.
- In case that the deployment is carried out in a real vehicle, the estimation of the states becomes essential. Although the objective of the project lies in the simulation, the state estimation module could be of great help to validate the vehicle model.
- Develop the system integration with inputs that are not given by a designed maneuver or a driver but by a trajectory tracking control, i.e SWA and torque request would be provided by the tracker.
- Implement combined slip conditions (longitudinal and lateral) in the tire model of the simulation plant.
- Test the controller performance under different motor configurations (two rear motors and one front motor, one rear motor and one front motor, etc.)
- Compare the current real-time performance with other nonlinear commercial solver such as ACADO Toolkit.

Bibliography

ACEA. The automobile industry pocket guide. https://www.acea.auto/files/ACEA_Pocket_Guide_2020-2021.pdf, 2021. [Accessed: 16-02-2022].

ADEME. Lca modeling and evaluation of consumer products and capital goods. https://bilans-ges.ademe.fr/documentation/UPLOAD_DOC_EN/index.htm?ordinateurs_et_equuipements_pe.htm, 2018. [Accessed: 07-06-2022].

A. Al-Hmouz, J. Shen, R. Al-Hmouz, and J. Yan. Modeling and simulation of an adaptive neuro-fuzzy inference system (anfis) for mobile learning. *IEEE Transactions on Learning Technologies*, 5(3):226–237, 2012. doi: 10.1109/TLT.2011.36.

J. Andersson. Vehicle dynamics: optimization of electronic stability program for sports cars. Bachelor's thesis, LuleåUniversity of Technology, 2008.

O. Barbarisi, G. Palmieri, S. Scala, and L. Glielmo. Ltv-mpc for yaw rate control and side slip control with dynamically constrained differential braking. *European Journal of Control - EUR J CONTROL*, 15:468–479, 12 2009. doi: 10.3166/ejc.15.468-479.

E. A. Baselga. *Advances in Planning and Control for Autonomous Vehicles*. PhD thesis, Universitat Politècnica de Catalunya, 2020.

BOE. Ley 27/2014, de 27 de noviembre, del impuesto sobre sociedades. https://www.boe.es/eli/es/l/2014/11/27/27/con, 2022. [Accessed: 10-06-2022].

M. Börner and R. Isermann. The characteristic velocity stability indicator for passenger cars. *Vehicle System Dynamics*, 43:601–612, 08 2005. doi: 10.1080/0042311042000266793.

V. Charitopoulos and V. Dua. Explicit model predictive control of hybrid systems and multiparametric mixed integer polynomial programming. *AIChE Journal*, 62, 06 2016. doi: 10.1002/aic.15396.

S. Chaubey. Slam of autonomous vehicles using the takagi-sugeno approach. Master's thesis, Universitat Politècnica de Catalunya, 2021.

W. Chen, H. Xiao, Q. Wang, L. Zhao, and M. Zhu. *Integrated vehicle dynamics and control*. John Wiley & Sons, Nashville, TN, Mar. 2016.

COIN-OR. Ipopt documentation. `https://coin-or.github.io/Ipopt/OPTIONS.html`. [Accessed: 25-02-2022].

L. De Novellis, A. Sorniotti, P. Gruber, and A. Pennycott. Comparison of feedback control techniques for torque-vectoring control of fully electric vehicles. *IEEE Transactions on Vehicular Technology*, 63(8):3612–3623, 2014. doi: 10.1109/TVT.2014.2305475.

M. Diehl, H. Bock, J. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12:577–585, 06 2002a. doi: 10.1016/S0959-1524(01)00023-3.

M. Diehl, H. Bock, J. P. Schlöder, R. Findeisen, Z. Nagy, and F. Allgöwer. Real-time optimization and nonlinear model predictive control of processes governed by differential-algebraic equations. *Journal of Process Control*, 12(4):577–585, 2002b. ISSN 0959-1524. doi: https://doi.org/10.1016/S0959-1524(01)00023-3. URL `https://www.sciencedirect.com/science/article/pii/S0959152401000233`.

D. Diez, E. Velenis, D. Tavernini, E. Smith, E. Siampis, and A. M. Soltani. Front/rear axle torque vectoring control for electric vehicles. *Journal of Dynamic Systems, Measurement, and Control*, 141, 11 2018. doi: 10.1115/1.4042062.

A. Domahidi and J. Jerez. Forces professional. Embotech AG, url=https://embotech.com/FORCES-Pro, 2014–2019.

Embotech. Forces pro user manual v5.1.0. `https://forces.embotech.com/Documentation/_static/FORCESPRO.pdf`, 2022. [Accessed: 09-03-2022].

Y.-S. Eom, J.-H. Hong, S.-J. Lee, E.-J. Lee, J.-S. Cha, D.-G. Lee, and S.-A. Bang. Emission factors of air toxics from semiconductor manufacturing in korea. *Journal of the Air Waste Management Association (1995)*, 56:1518–24, 12 2006. doi: 10.1080/10473289.2006.10464556.

J. Frasch, A. Gray, M. Zanon, J. Ferreau, S. Sager, F. Borrelli, and M. Diehl. An auto-generated nonlinear mpc algorithm for real-time obstacle avoidance of ground vehicles. pages 4136–4141, 07 2013. doi: 10.23919/ECC.2013.6669836.

M. K. Ghezzi. Control of a four in-wheel motor drive electric vehicle. Master's thesis, Universitat Politècnica de Catalunya, 2017.

S. Gros, M. Zanon, R. Quirynen, A. Bemporad, and M. Diehl. From linear to nonlinear mpc: bridging the gap via the real-time iteration. *International Journal of Control*, 93:1–19, 09 2016. doi: 10.1080/00207179.2016.1222553.

W. Grzegoek and K. Weigel-Milleret. Torque vectoring for improving stability of small electric vehicles. *IOP Conference Series: Materials Science and Engineering*, 148: 012008, 09 2016. doi: 10.1088/1757-899X/148/1/012008.

A. Haddoun, M. Benbouzid, D. Diallo, R. Abdessemed, J. Ghouili, and K. Srairi. Sliding mode control of ev electric differential system. In *ICEM'06*, page 6pp, 2006.

S. Inoue, T. Ozawa, H. Inoue, P. Raksincharoensak, and M. Nagai. Cooperative lateral control between driver and adas by haptic shared control using steering torque assistance combined with direct yaw moment control. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 316–321, 2016. doi: 10.1109/ITSC.2016.7795573.

R. Isermann. *Automotive control*. ATZ/MTZ-Fachbuch. Springer, Berlin, Germany, 1 edition, Sept. 2021.

ISO Copyright Office. Passenger cars – test track for a severe lane-change manoeuvre – Part 1: Double lane-change. Standard ISO 3888-1:2018(E), International Organization for Standardization, Geneva, CH, 2016. URL `https://www.iso.org/standard/67973.html`.

K. Jalali, T. Uchida, J. McPhee, and S. Lambert. Integrated stability control system for electric vehicles with in-wheel motors using soft computing techniques. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems*, 2(2009-01-0435): 109–119, 2009.

X. Jin, G. Yin, and N. Chen. Advanced estimation techniques for vehicle system dynamic state: A survey. *Sensors*, 19(19), 2019. ISSN 1424-8220. doi: 10.3390/s19194289. URL `https://www.mdpi.com/1424-8220/19/19/4289`.

J. Kang, Y. kyongsu, and H. Heo. Control allocation based optimal torque vectoring for 4wd electric vehicle. *SAE Technical Papers*, 04 2012. doi: 10.4271/2012-01-0246.

D. Kasinathan, A. Kasaiezadeh, A. Wong, A. Khajepour, S.-K. Chen, and B. Litkouhi. An optimal torque vectoring control for vehicle applications via real-time constraints. *IEEE Transactions on Vehicular Technology*, 65(6):4368–4378, 2016. doi: 10.1109/TVT.2015.2467374.

J. Löfberg. Yalmip : A toolbox for modeling and optimization in matlab. In *In Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

Q. Lu, P. Gentile, A. Tota, A. Sorniotti, P. Gruber, F. Costamagna, and J. De Smet. Enhancing vehicle cornering limit through sideslip and yaw rate control. *Mechanical Systems and Signal Processing*, 75, 12 2015. doi: 10.1016/j.ymssp.2015.11.028.

A. Lucchini, S. Formentin, M. Corno, D. Piga, and S. Savaresi. Torque vectoring for high-performance electric vehicles: An efficient mpc calibration. *IEEE Control Systems Letters*, PP:1–1, 03 2020. doi: 10.1109/LCSYS.2020.2981895.

I. Malmgren. Quantifying the societal benefits of electric vehicles. *World Electric Vehicle Journal*, 8(4):996–1007, 2016. ISSN 2032-6653. doi: 10.3390/wevj8040996. URL `https://www.mdpi.com/2032-6653/8/4/996`.

MathWorks. Coder.extrinsic documentation. `https://es.mathworks.com/help/simulink/slref/coder.extrinsic.html`, 2022a. [Accessed: 07-04-2022].

MathWorks. Specify constraints. `https://es.mathworks.com/help/mpc/ug/specifying-constraints.html#responsive_offcanvas`, 2022b. [Accessed: 11-02-2022].

MathWorks. Solve nonlinear curve-fitting (data-fitting) problems in least-squares sense. `https://es.mathworks.com/help/optim/ug/lsqcurvefit.html`, 2022c. [Accessed: 27-04-2022].

MathWorks. Variant subsystems. `https://es.mathworks.com/help/simulink/ug/variant-subsystems.html`, 2022d. [Accessed: 24-03-2022].

V. Mazzilli, S. De Pinto, L. Pascali, M. Contrino, F. Bottiglione, G. Mantriota, P. Gruber, and A. Sorniotti. Integrated chassis control: Classification, analysis and future trends. *Annual Reviews in Control*, 51, 04 2021. doi: 10.1016/j.arcontrol.2021.01.005.

McKinsey. Disruptive technologies: Advances that will transform life, business, and the global economy. `https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/disruptive-technologies`, 2013. [Accessed: 16-02-2022].

M. Metzler, D. Tavernini, P. Gruber, and A. Sorniotti. On prediction model fidelity in explicit nonlinear model predictive vehicle stability control. *IEEE Transactions on Control Systems Technology*, PP, 10 2020. doi: 10.1109/TCST.2020.3012683.

W. F. Milliken and D. L. Milliken. *Race Car Vehicle Dynamics*. Premiere Series. SAE International, Warrendale, Dec. 1995.

D. Miloradovic, J. Glišović, N. Stojanović, and I. Grujić. Simulation of vehicle's lateral dynamics using nonlinear model with real inputs. *IOP Conference Series: Materials Science and Engineering*, 659(1):012060, Oct. 2019. doi: 10.1088/1757-899x/659/1/012060. URL `https://doi.org/10.1088/1757-899x/659/1/012060`.

M. Nikowitz. *Advanced Hybrid and Electric Vehicles: System Optimization and Vehicle Integration*. 01 2016. ISBN 978-3-319-26304-5. doi: 10.1007/978-3-319-26305-2.

H. Pacejka. *Tire and Vehicle Dynamics*. Butterworth-Heinemann, hardcover edition, 4 2012. ISBN 978-0080970165.

A. Parra, A. Zubizarreta, J. Pérez, and M. Dendaluce. Intelligent torque vectoring approach for electric vehicles with per-wheel motors. *Complexity*, 2018, 02 2018. doi: 10.1155/2018/7030184.

A. Parra, D. Tavernini, A. Sorniotti, P. Gruber, A. Zubizarreta, and J. Pérez. On nonlinear model predictive control for energy-efficient torque-vectoring. *IEEE Transactions on Vehicular Technology*, PP, 09 2020. doi: 10.1109/TVT.2020.3022022.

P. Perinciolo and E. Sondhi. Model based handling analyses, 2018.

L. Peters. Model predictive control - part 1: Introduction to mpc. `https://www.youtube.com/watch?v=XaD8Lngfkzk&t=29s`, 2021. [Accessed: 03-03-2022].

R. Rajamani. *Vehicle Dynamics and Control (Mechanical Engineering Series)*. Springer, 12 2011.

M. Salehi, J. Noordermeer, L. Reuvekamp, and A. Blume. Understanding test modalities of tire grip and laboratory-road correlations with modeling. *Tribology Letters*, 69, 09 2021. doi: 10.1007/s11249-021-01490-2.

B. Schofield and T. Hägglund. Optimal control allocation in vehicle dynamics control for rollover mitigation. pages 3231 – 3236, 07 2008. doi: 10.1109/ACC.2008.4586990.

B. Schofield, T. Hägglund, and A. Rantzer. Vehicle dynamics control and controller allocation for rollover prevention. pages 149 – 154, 11 2006. doi: 10.1109/CACSD-CCA-ISIC.2006.4776639.

E. Siampis, E. Velenis, S. Gariuolo, and S. Longo. A real-time nonlinear model predictive control strategy for stabilization of an electric vehicle at the limits of handling. *IEEE*

Transactions on Control Systems Technology, PP:1–13, 10 2017. doi: 10.1109/TCST.2017.2753169.

SimWB. About simulation workbench (simwb). https://wiki.simwb.com/wiki/Main_Page, 2022. [Accessed: 24-05-2022].

SWEEP. Economic and air quality benefits of electric vehicles in nevada. https://energy.nv.gov/uploadedFiles/energynvgov/content/Programs/SWEEP_Economic_and_AQ_Benefits_of_EVs_in_NV-Sept_2014.pdf, 2014. [Accessed: 27-05-2022].

K. Tse. Ota 2018 debrief: Understeer gradient test. https://kktse.github.io/jekyll/update/2018/12/01/understeer-gradient-identification.html, 2018. [Accessed: 18-04-2022].

VI-GRADE. *VI-CarRealTime 20.2 Documentation*. VI-GRADE, Darmstadt, Germany, 2 2021.

G. Vorotović, B. Rakicevic, S. Mitić, and D. Stamenković. Determination of cornering stiffness through integration of a mathematical model and real vehicle exploitation parameters. *FME Transactions*, 41:66–71, 01 2013.

A. Zanelli, A. Domahidi, J. Jerez, and M. Morari. Forces nlp: an efficient implementation of interior-point... methods for multistage nonlinear nonconvex programs. *International Journal of Control*, pages 1–17, 2017.

M. Zanon, J. Frasch, M. Vukov, S. Sager, and M. Diehl. *Model Predictive Control of Autonomous Vehicles*, volume 455. 03 2014. ISBN 978-3-319-05370-7. doi: 10.1007/978-3-319-05371-4_3.

ZePerfs. Bmw 320 da (g20) 190 cv data sheet. https://zeperfs.com/es/fiche7860-bmw-320-da.htm#Caracter%C3%ADstica, 2021. [Accessed: 16-12-2021].