Universidad de Alcalá
Escuela Politécnica Superior

Grado en Ingeniería en Electrónica y Automática Industrial

Trabajo Fin de Grado

Comparative analysis of MPC controllers applied to Autonomous Driving

Autor: Carmen Barbero Ruiz

Tutor: Iván García Daza

2021

UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería en Electrónica y Automática Industrial

Trabajo Fin de Grado

Comparative analysis of MPC controllers applied to Autonomous Driving

Autor: Carmen Barbero Ruiz

Director: Iván García Daza

“Sólo si nos detenemos a pensar en las pequeñas cosas llegaremos a comprender las grandes.”

José Saramago

Acknowledgements

This work represents the end of a period of hard work and perseverance. The hours dedicated to this project - half joyful, half tearful - have brought me an important personal enrichment that I will never forget. Throughout the development of this thesis, I have received a great deal of support and assistance. To a greater or lesser extent, I would like to thank everyone who has been by my side.

I cannot begin to express my thanks to my professor, Iván, for the help and guidance he has given me during this process. Thank you for giving me the opportunity to participate in this project.

To my family, thank you for believing in me, even when I did not. Your support and patience is reflected here.

To my friends, thank you for all the moments, advice and good times.

And to the ones who are no longer with us, all your efforts have led me here. Wherever you are, this is especially for you.

Resumen

Este trabajo presenta el diseño de un sistema de evasión de obstáculos, aplicable en situaciones de emergencia. La solución propone un MPC multivariable para controlar la posición, orientación y velocidad del vehículo autónomo. El controlador considera las limitaciones físicas del vehículo, así como la morfología de la vía para conseguir minimizar los posibles daños que puedan afectar al sistema y en consecuencia a la pérdida de control del vehículo. Las restricciones principales están basadas en las fuerzas laterales que afectan a los neumáticos, obtenidas de la implementación de los modelos cinemático y dinámico de la planta. Inicialmente, el controlador hace que el sistema siga una trayectoria predefinida. No obstante, tomará las acciones de evasión necesarias cuando detecte obstáculos, para conseguir realizar trayectorias libres de colisiones. Los resultados obtenidos tras la validación del sistema se presentan con el simulador para conducción autónoma CARLA.

Palabras clave: Model Predictive Controller, Vehículos Autónomos, Evasión de Obstáculos, CARLA.

Abstract

This work presents the design of an obstacle avoidance system, employable in emergency situations. The solution proposes a multivariable Model Predictive Controller (MPC) to control the position, orientation and velocity of an autonomous vehicle. The controller considers the vehicle′s physical limitations, as well as the road morphology, to minimize any possible damage to the system and the loss of control of the vehicle. Its main constraints are based on the lateral tire forces, obtained from the implementation of a kinematic and dynamic plant model. The controller, initially following a predefined trajectory, will take the needed evasive actions in order to perform a collision-free trajectory, in case of an obstacle detection. The results obtained from the system validation are presented with CARLA open-source simulator for autonomous driving.

Keywords: Model Predictive Controller, Autonomous Vehicles, Obstacle Avoidance, CARLA.

Contents

List of Figures

List of Tables

List of source code listings

This work has been totally implemented in Python. The source code is available at the following repository:

GitHub repository or go to the next url: `https://github.com/ivangarciad/mpcavoidance.git`

List of Acronyms

ACEA	European Automobile Manufacturers Association.
API	Application Programming Interfaces.
CARLA	Car Learning to Act.
CG	Centre of Gravity.
COBYLA	Constrained Optimization BY Linear Approximation.
GPS	Global Positioning System.
GPU	Graphics Processing Unit.
ITSS	IEEE Intelligent Transportation Systems Society.
IV21	32nd IEEE Intelligent Vehicles Symposium.
LIDAR	Laser Imaging Detection and Ranging.
MIMO	Multiple-Input Multiple-Output.
MPC	Moderl PRedictive Controler.
PCB	Prediction and Cost-function Based.
SDRE	State Dependent Riccati Equation.
SISO	Single-Input Single-Output.
V2V	Vehicle to Vehicle.

Chapter 1

Introduction

1.1 Motivation

According to the World Health Organization [7] approximately 1.35 million people die every year due to traffic accidents. Despite all the technological advances and legislative improvements done to reduce these alarming numbers, in the end the security of the users on the road depends only on the own drivers. However, the increasing automation in the vehicle industry aims to reduce, if possible, the workload that is normally suffered by the drivers. In this way, the possible human errors that can occur are minimized, as well as the probability of an accident to occur.

In 2015, the General Dependencies of Traffic of Spain released an administrative instruction (15/V-113) [8] in order to define what an autonomous vehicle is. According to this instruction, an autonomous vehicle is *every vehicle with motor capacity, equipped with technology that allows its driving with no active control or supervision of a driver, even if the autonomous technology is active or inactive, temporarily or permanently.* These vehicles are ruled by different types of technology, as radars, cameras, ultrasound systems or even radio antennas. All these methods provide a different level of autonomy, and all combined can achieve a more reliable and robust system. However, the technology implemented in each process of autonomous driving ought to be previously studied and analyzed in detail in order to fulfill the system's requirements.

Over the last 40 years, this field of study has suffered a significant evolution, due to the increment in the power and the reduced cost of the sensors and computer systems. The increasing automation in the vehicle industry aims to reduce, if possible, the workload that is normally suffered by the drivers. This will not only provoke improvements in the driver's economy and travel time, but also the drastic reduction of traffic deaths.

In order to bring new advances and contributions to this field of study, numerous specialists in the field are developing techniques related to trajectory planning and decision making. Among others, one of the most relevant areas of study in this field is the **obstacle avoidance problem**: a task similar to the navigation problem, whose result is a collision free motion. This work is centered in the latter: the implementation of a solution for the obstacle avoidance problem, given by solutions proposed by a position controller.

1.2 Objectives and organization of the project

The main objective of this project is to design a controller for an autonomous vehicle. By acting over the throttle and steering wheel of the vehicle, the complete autonomy from the user is achieved. The controller chosen to do so is a Model Predictive Controller (MPC), in order to control the position and orientation of the vehicle. Additionally, an important point of this project is the development of an algorithm in order to resolve any obstacle avoidance problem. This algorithm provides a safety capability of the controller. In order to deliver the optimal control action (follow a reference trajectory, with no collisions), the controller considers avoidance constraints (lateral distance to the obstacle, road morphology) and safety constraints for the manoeuvre (forces acting on the tires, slips).

The flexibility of the combination of both concepts inside the controller will allow its application in multiple scenarios, whether the vehicle undergoes a collision free trajectory or there's a certain obstacle in its proximities.

For simplification purposes, this work has been divided in several stages: a theoretical introduction to the field of study, the design of the MPC controller and the obstacle avoidance problem approach. The organization of this work is divided in the following chapters.

1. **Autonomous driving.** State of the art and theoretical approach to the most popular techniques and fields of work present in autonomous driving.
2. **Model Predictive Control.** Theoretical introduction to this field of study. This chapter details the model of the vehicle implemented in the controller (for the prediction stage), as well as the cost function and constraints which determine the final control action.
3. **Obstacle Avoidance Problem.** Theoretical approach to this problem, in order to obtain a collision-free motion of the vehicle. The algorithm proposed as a solution, including its main safety characteristics, are presented.
4. **Results.** With the help of CARLA open-source software, the validation of this process is virtually tested before the integration in any real system. The results of the simulations are presented, as well as the comparative analysis of them.
5. **Conclusions.** A brief analysis of the controller is finally presented, as well as possible future improvements and lines of work.

Chapter 2

Theoretical Approach

2.1 State of the Art

Modern times bring continuous advances in the field of autonomous driving. Over the last 40 years, numerous research studies have explored different ways of controlling unmanned vehicles. According to S. C. Pendleton et al [9], autonomous driving consists of three different modules that, when combined properly, are capable of interacting with the environment in order to govern the motion of the vehicle. A **perception module** gathers through sensors the localization and environmental perception, in order to transfer to a **planning module** the model and vehicle's pose. This module, depending on the aim of the project, will deliver the corresponding target actions to the **control module**. The latter commands, needed to perform either a path or a trajectory tracking, will be given to the vehicle actuators.

The module in charge of the **automatic control** is the most important step, due to its mission: guarantee the motion of the vehicle. From a model-based approach, the control module is defined by the **type of control** made (longitudinal, lateral or integrated) and the **design model** used (kinematic, dynamic or a combination of both). It is important to emphasize that the control module is built up mainly by a controller. In this way, different studies may obtain different results, depending on the control aim and needs each research may have. Thus, E. Watcher[10] proposed a *feedback-feedforward steering controller*, to minimize the error in the lateral path tracking when the system works in limit handling conditions. In this research, by using a combined design model and the SDRE control technique, a controller with a high reliability and good performance was obtained. Also, K. Zhao et al [11] proposed a *nonlinear model predictive control optimization algorithm*, in order to reduce the errors in the lateral trajectory tracking. Here, they considered the use of a kinematic model and of the collocation method: a method to solve optimal control problems, usually appropriate when optimizing trajectories and/or parameters. In this way the calculation load is reduced, and the algorithm obtained suits the real-time conditions needed.

From what has been mentioned, it is clear to see that control types differ in the field of autonomous driving. However, one of the most popular control techniques in automotive industry applications today is what is known as **Model Predictive Control** (MPC). MPC is a very versatile and competitive algorithm: it can be easily used in different levels of the process control, as well as work with constraints within the control process. It can be applied in SISO and MIMO systems, as well as in both linear and nonlinear processes. However, the fact that the feedback control algorithm has a **model-based approach** directly affects its performance. Being dependent on the model of the system working with, makes the MPC performance to be subordinated to the accuracy and the complexity of this model. The closer the model is to the real system, the better the results the MPC will provide. From this, it can be understood that there are different approaches when applying this control algorithm. Different vehicle models, constraints, output and control variables, among others, result in completely different controllers and, therefore, results.

In this paper, an MPC algorithm for the **lateral control** of an autonomous vehicle is proposed. One of the main objectives is the generation of a control law which considers the cartesian coordinates of the vehicle, its heading angle and velocity as state parameters, plus the vehicles throttle and steering wheel angle for the control action. As F. Kühne et al [12], the non-linear model dynamics implemented in this research are linearized, in order to speed up the optimization process of the algorithm. However, in contrast with the latter, in this paper the MPC control process of the autonomous vehicle differs in the cost function implemented, due to the parameters considered. The **cost function**, as denoted by E.F. Camacho et al[2], is colloquially defined as *where the future tracking error is considered.* This function may consider only the future errors, but also the control effort of the process. In contrast with Kühne, this research has included more effort parameters inside the cost optimization problem. Specifically, the cross-error track implemented considers a linearized model ignored by Kühne, but used in many open source algorithms, like T. Raack[13] and F. Pucher[14].

Furthermore, several **obstacle avoidance conditions** are included in the optimization problem. In literature, this collision avoidance of the autonomous system is usually done, as mentioned by B. Switzer[15], with a replanning strategy. If an obstacle is encountered within the initial trajectory, the system will begin again and the planner will hypothesizes an alternate path. This process continues until a successful path is found. Without the recalculation of the vehicle's trajectory, but with the help of constraints inside the control problem, the control of the autonomous vehicle in this project executes safe trajectories with no collisions. By determining a safe region through which the vehicle ought not to pass, the MPC provides a trajectory solution which fulfills both minimum error condition and obstacle avoidance condition.

Likewise, conclusions obtained by J. Kong et al[16] and L. Tang et al[17] have also been heeded when doing this research. The use of a simple model, like the kinematic bicycle model implemented by J. Kong, benefits the computational effort of the model: simplicity means less computations. Besides, it makes the algorithm more satisfactory and suitable in a wide range of experiments. However, in contrast to L.Tang, the model implemented for the trajectory tracking does not include a slip compensation (to avoid the undesired complexity previously mentioned in the final algorithm).

2.2 Autonomous Driving

During the last years, the field of autonomous driving has suffered a great number of improvements, due to the increment in the power and the reduced cost of the sensors and computer systems. These two factors have provoked the significant evolution of the automation level of vehicles.

The three main lines of action of the development of software systems for autonomous driving are centered in the **perception**, **planning** and **control** of the system. The term **perception** is centered in making the system work, based on the information obtained from its surroundings, either the detection of a signal or an obstacle, or the location the system can make of itself (its position with respect to the environment). **Planning** aims to take decisions based on the main objectives of the system; that is to say, it focuses on optimizing all the decisions, for moving the system from an initial state to a final one in the best way possible. Finally, **control** refers to the execution of actions already planned, generated by higher level processes.

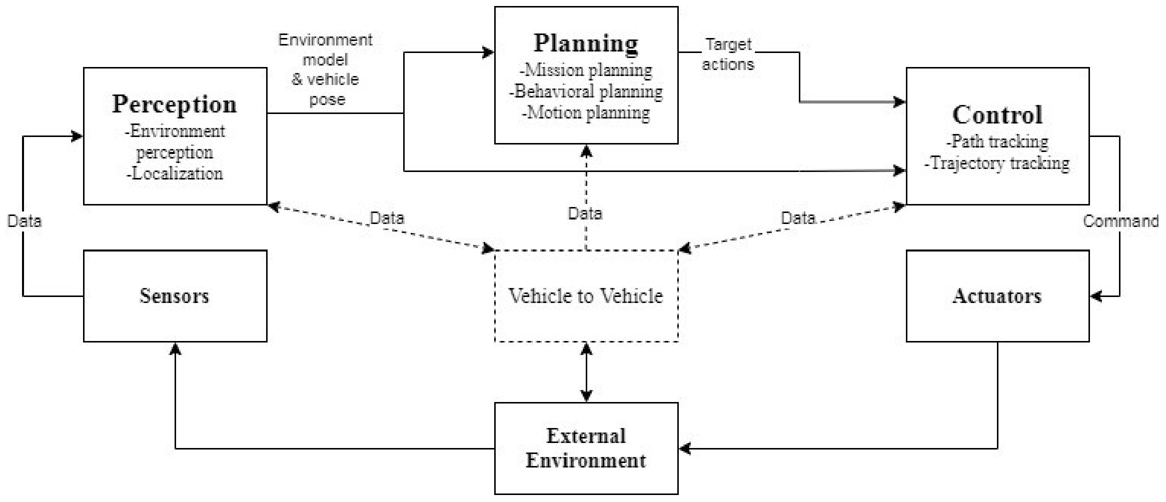


Figure 2.1: Autonomous Driving main lines of action.

2.2.1 Perception

The **perception module**, as it has been previously mentioned, is the system created to work with the sensory signals coming from the vehicles environment. Its two submodules collect the information from the surroundings in order to handle its detection and identification, for its use by the next modules. They also determine the position of the vehicle.

One of the two submodules inside the perception block is called the **environment perception module**. With the help of the sensors already included in the system, it notices what is happening in the environment by using the information gathered by cameras, LIDARs or a combination of both. All the systems mentioned have two critical elements: to obtain the roads path, and to detect the elements on it. The systems of vision centered in the perception of the environment are classified into two types: road detection and detection of objects on the road. The *road detection* is centered in the exhaustive study of the road, mainly in the creation of a representative model of the desired path. The *detection of objects on the road* is centered in the detection and classification of objects and pedestrians that may be located on the road. There are lots of different methods to do this, but the most used one is Deep Learning.

The second submodule in the perception block is the **localization module**, which deals with the determination of the vehicles position and motion. It is generally known as a *pose estimation prob-*

lem. It can be subdivided into two different fields of study: pose fixing problem (position defined by an algebraic equation), and dead reckoning problem (prediction of measurements given by differential equations). Generally, the most extended method for performing autonomous vehicles localization are the GPS satellite navigation system (problematic in areas with unreliable service signals, like tunnels) and map aided localization algorithms.

2.2.2 Planning

As the field of study of autonomous vehicles has been developed, it has been proved that the more exhaustive the planning, the better the results of autonomous driving. Over time, the most extended planner in autonomous vehicles has been a three-level planner, consisting on a mission level (high level objectives), a behavioral level (decision maker, creates the local objectives) and a motion level (decision maker, for fulfilling the local objectives).

When talking about **mission planning**, this module is centered into working with high level decisions, basically driving missions which are translated into lane-level submissions. In other words, the process of going from an initial to a final position is transformed into lane processes, of deciding which lane the vehicle should be in in order to do so, etc. Also, it deals with the execution of goals: depending on the environment conditions, the module decides if the next decision can be executed or not (i.e if theres a stop signal, the car ought to stop, therefore the next submission of moving must first be checked by the perception system (is it safe to move, or must the system wait?)).

Behavioral planning is centered into the possible traffic, either static or dynamic, present on the road where the vehicle is. The input for this module are all the moving references and road blockages present in the environment; the output are the indications for the controller, in order to control the lateral movement of the vehicle, speed, and others. For doing so, a prediction and cost-function based algorithm (PCB) is used: from all the possible strategies to plan the behavior of the system, it predicts based on one of them and evaluates with cost function algorithm.

The term **motion planning** refers to the generation of the vehicles trajectory, considering the results obtained from the behavioral planning (depends on the dynamic parameters and the behavioral output), and based also on the efficiency of the computational system. Some of the main techniques used in motion planning are *combinatorial planning* (algorithm which does not use approximations: if it does not find a solution, it notifies that it does not exist, but it does not resort to approximations) and *sample-based planning* (it tolerates probabilistic and approximated solutions).

2.2.3 Control

When talking about autonomous vehicles, control is the study of managing dynamical systems (mainly its motion) in order to execute the specific tasks, with the help of suitable control actions. For this, some input signals will command the system, in order to generate output actions that will make the system perform the wanted motions. The module that performs this process is known as the **controller**: it relates the environment signals into indicators which can be understood by the system. Generally, most controllers are ruled by **feedback control**: a structure where the output signal is used in order to control the input signal, for achieving a compensated future response. This is done in order to achieve a system that can compensate itself, in case of deviations and changes caused by unwanted errors and disturbances. The restrictions of feedback controllers are related to different factors, like delays (the response to errors can only be done when the error occurs) and coupled responses (the response to errors and disturbances is done by the same system than the response to a normal reference).

In relation to the generation and tracking of the vehicles movement, this can be done in two different ways. When knowing the path information, the first approach is to use an optimization method to generate and track at the same time. The second approach is to decouple both processes, and work independently with them. Regarding the latter, there are different methods applied in this field; some of them are mentioned in the following lines:

- **Trajectory Generation.** Its goal is to find an input control signal u(t) which associates to a specific state trajectory x(t) of the system, for a given time interval. It is done with the help of robotics (mainly sensors) and taking advantage of the vehicle dynamics. Nonetheless, a combination of both techniques should be done in order to optimize the results of the method.

- **Trajectory Tracking.** Either by applying geometric or model-based methods, the objective of this procedure is to perform the movement from an initial point A to a final point B, taking into consideration the velocity of the system. In terms of geometric methods, there is a wide variety of algorithms in order to perform the tracking of the trajectory, like the Pure Pursuit path tracking Algorithm (easy to implement and robust to disturbances and errors) and the Stanley method (better results, but less robust). On the other hand, model-based methods need the path to be continuous and are not as robust as the others; they differentiate internally according to the speed of the system. For slow speeds, there are kinematic models and linearized dynamics models for vehicles. Also, there are **Model Predictive techniques**, like MPC, used. By using high computational power systems, MPC can solve the possible non-linearities present when talking about trajectory tracking.

2.2.4 Cooperation with the Vehicle, V2V

Control of autonomous vehicles is also done with the help of **vehicular communication**: being able to gather information given by different vehicles can provide the estimation of the future states the system will have. With this, a better future navigation of the autonomous vehicle is achieved. Here, the three main frameworks are centered in the *communication among vehicles* (by wireless connection), a *cooperative localization* (creation of maps from the perceived information) and *motion coordination* (the share of future trajectories allows the prediction of changes in the environment).

Thus, **vehicle platooning** is commonly used among self-driving cars' communication. According to ACEA [18] and C. Bergenhem et al [19], this concept links two or more vehicles with the help of connectivity technology and driving support systems, in order to react and adapt to motion changes. The V2V communication system implemented is able to share the needed data between the vehicles, like the speed and the control of the brake, as depicted in figure 2.2.

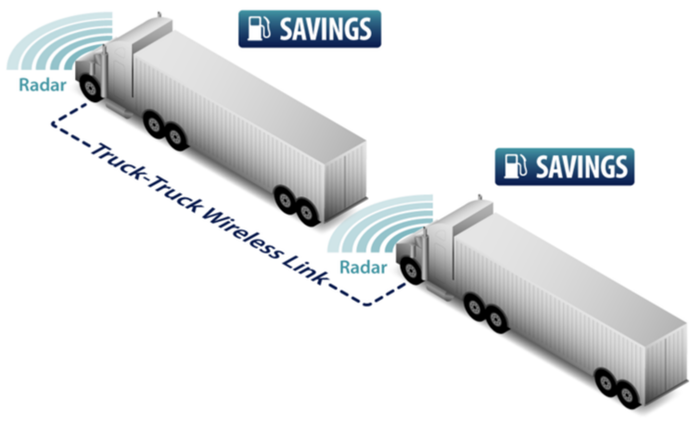


Figure 2.2: Truck platooning. Source [1].

Chapter 3

Model Predictive Control

3.1 Introduction

As it has been mentioned, autonomous driving consists of three major fields of study: localization, perception and control. The main objective of this last field, **control** , is not only knowing how to govern the vehicle to follow a reference trajectory; it is also an objective to perform this trajectory in the smoothest way possible. In control terms, the process of path tracking can be considered a **multi-constrained optimization problem**. Thereby, this problem needs to consider not only the constraints of the location error, but also the constraints related to the mechanical and electrical parts of the vehicle. To do so, one of the most popular advance techniques used in industrial process applications is **Model Predictive Controller** (MPC). The design formulation of the MPC is ruled by a multivariable system structure that controls determined performance parameters, which directly govern certain engineering aspects of the system.

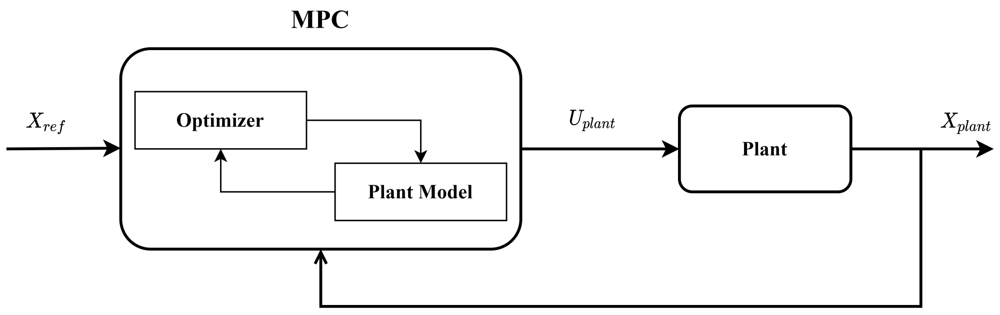


Figure 3.1: MPC General Schematic.

3.2 General Aspects

The execution of the MPC relies on an algorithm which takes the value of the existing output parameters of the process, in order to make predictions of its future outputs. By doing so, the algorithm can anticipate itself in order to perform the needed control actions. This controller works with a **multivariable cost function** under future reference state conditions: it minimizes this function with respect to multiple constraints. The main objective of the controller is to calculate a series of control sequences in the prediction horizon. With this, it makes the error between the output and the reference to be reduced as much as possible. When the MPC is used for path tracking, it minimizes the interval between the reference path and the vehicle's predicted trajectory, by solving a quadratic programming problem with multiple constraints.

A simple way to explain what any MPC does is to compare its procedure to when a certain user drives a car [20]. The user knows how the car works and the road he may be driving through. In order to follow the road, the user must simulate in his head the actions the car must perform. This way of predicting the future trajectory, is what will make the user act on any car element for achieving the wanted trajectory. Consequently, the different parameters conforming the MPC can be associated with different car parameters.

- **Prediction model.** It describes how the vehicle is expected to move on the map.
- **System constraints.** Set of rules which govern the driving of the vehicle, considering the road driving jurisdiction and the mechanical capabilities of the system. Some of these constraints are *velocity*, *acceleration* and the *steering angle* of the vehicle.
- **Disturbances.** They cause an *uncontrolled deviation* from the wanted trajectory.
- **Setpoint** Desired location, the reference trajectory.
- **Cost function (J).** A goal of minimum time, minimum distance, etc.
- **Receding horizon (N).** It is used to plan the trajectory of the car and the driver actions periodically. In this time, the overall set of actions over a time horizon N is found, but only the first one is used. When done, the states are updated, and the process is planned again for the next step. It corresponds to the simulation of the possible future actions the user can think about, as in figure 3.2.

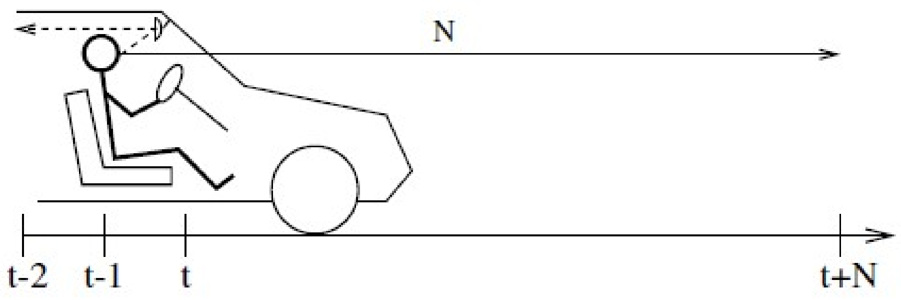


Figure 3.2: Analogous representation of the receding horizon in a driver. Source [2].

MPC is a very versatile and competitive algorithm: it can be easily used in different levels of the process control, as well as work with constraints in the control process. The main properties which make this controller so adaptable are:

- **Prediction**. Before planning, the consequences and possibilities of the activity must be considered in order to obtain an appropriate output. In driving terms, the user must predict the safe braking distance for the vehicle in order to avoid a crash. If this is not done, there is no guarantee that there will be no problems. This commonsense analysis of the process must be also applied to autonomous driving.

- **Model-based approach.** The controller needs a mathematical model close to the real model of the system, in order to perform the predictions and actions of the control system with little or no error. Being dependent on the model of the system makes the performance to rely on the accuracy and the complexity of the latter. The model used in order to design the controller must be **ideally linear** (to form predictions easily and with accurate results) plus have parameters which are easily recognizable. The closer the model is to the real system, the better the results the MPC will provide.
- **States.** It is important to note that not every single control strategy (for instance, classical controllers) allow to consider inside the design process the states of the vehicle. However, MPC can combine inside of the design different objective functions. In this way, it can look forward to ensuring safety and control.
- **Rolling optimization.** The optimization process periodically updates its input data information (its predictions and decisions), in order to obtain real time predictions (the most recent measured data). In this way, the prediction horizon of the autonomous vehicle will always refer to its current position, meaning the horizon will recede/move away from the user as this one moves. This receding horizon concept directly deals with the computational effort: its preview capability (knowing in advance the future reference in order to perform correctly the control signal) makes of MPC a widely used controller; however, in order to do so a very powerful processor is needed.
- **Feedback.** It is directly related to the use of the receding horizon: the evaluation of the vehicle's situation by the controller must be done periodically, with the help of feedback. Feedback corrects the possible modeling errors, which is especially useful for long range predictions.
- **Multi-variable systems.** One of the most important features of MPC is the fact that it can properly handle MIMO systems, which stand out for having multiple input and output signals. These systems are not easy to work with, since if their input and output signals interact and affect each other, the control process becomes complicated.
- **Constraint Handling.** The theory which governs Predictive Control includes this concept in the strategy development. It simply refers to the selection of the optimal trajectory only when the constraints of the system, either physical or safety-related, are satisfied. Constraints ensure that the strategies used are optimized, but also that they are safe according to the implemented model. Thus, the driving legislation determines minimum and maximum velocities for the road, which are constraints for the controller. Also, the turning angle range for the vehicle's steering wheel is also a constraint.

The strategy method of the MPC is developed in the following lines. Briefly, a model like the one shown in figure 3.1 obeys the following process:

1. A model representing the dynamic process of the system is used in order to predict the behavior of the plant and its future outputs. This output signal (3.1) is based on both past and current input and output measurements, taken up to instant k. Its calculation is done at instant k, for the different instants (k+i) of the prediction horizon (Hp).

$$y(k+i|k), fori = 1, ..., Hp \tag{3.1}$$

   Also, the following future control signal is considered.

$$u(k+i|k), fori = 0, ..., Hc-1 \tag{3.2}$$

As with the output, the nomenclature denotes that this signal is obtained by the calculation done at instant k, for the different instants (k+i) of the control horizon (Hc).

2. The control signal (3.2), is obtained by optimizing a cost function, which encompasses the different errors which must be considered in order to follow the reference trajectory. The cost function corresponds to equation (3.3).

$$J = \sum_{i=1}^{Hp} e^T Q_e e + \sum_{i=0}^{Hc-1} \Delta u^T Q_u \Delta u \tag{3.3}$$

That is to say, (3.3) contains the error between the reference trajectory (desired one) and the predicted output obtained by the controller (real one). By minimizing the error, the most accurate solution given by the controller is obtained.

3. From the solution obtained, only the first control signal of the receding horizon is implemented: $u(k|k)$, the rest are neglected. This decision is based on the fact that when implementing this control signal, the output in the next sampling instant, $y(k+1)$, is already known (the rejected control signals in this instant are already incongruous).

4. The first step is repeated with the values obtained from the previous iteration, and all the sequence is updated.

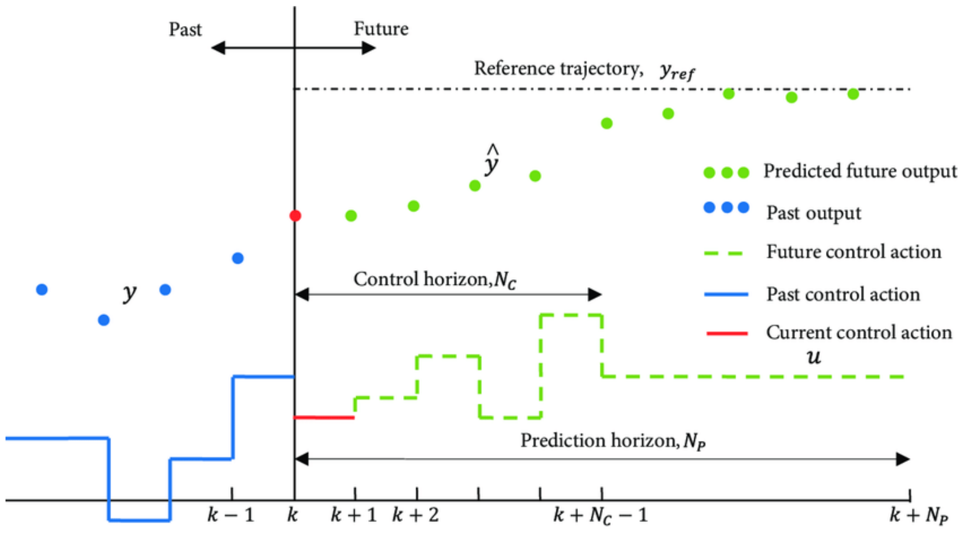


Figure 3.3: MPC Controller time response. Source [3].

This type of optimal control [21] aims to determine the optimal control input $u(k)$, for the plant, in order to drive the system onto a reference signal $r_{ref}(k)$ value, as well as minimizing a performance index J. Therefore, the required components in order to formulate this optimal control system applied to an autonomous vehicle, are a **model** of the plant that is controlled, a **performance index**, and certain **constraints**. The main components used in this project are described in the following pages.

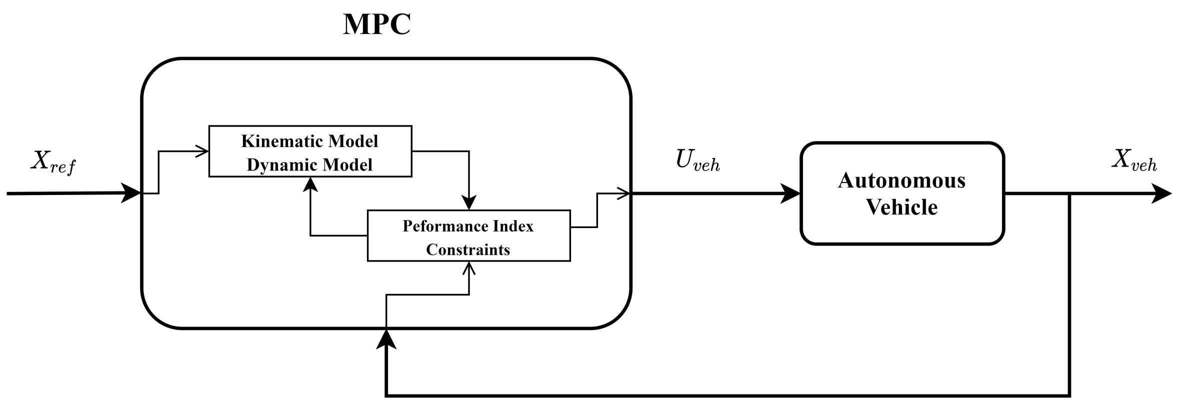


Figure 3.4: MPC application schematic.

3.3 Prediction Model

The behaviour of MPC controllers, as its name denotes, is dependant on the model of the plant that is controlled. In this paper, both dynamic and kinematic models are used to obtain the velocity, position and orientation of the vehicle. The implemented model uses the control actions, steering angle and throttle position, to evaluate the different states. The state vector of the system is defined in equation 3.4

$$X_{veh} = [x, y, \psi, v] \tag{3.4}$$

In the following lines, an analysis of the different models is presented.

3.3.1 Kinematic Model

When designing the Model Predictive Controller for an autonomous vehicle, the model of the vehicle representing the plant is needed. Generally, the simplest and most direct forward way to obtain it is to implement its **kinematic model**: a simple model which takes into consideration the speed, position and heading but neglects dynamic elements like forces, gravity and the vehicle's mass. Despite being a **simple model**, its simplicity makes of it a very versatile and computationally simple design. The kinematic model of a vehicle is basic enough to reduce the computational effort of the system working with, which makes the execution in real-time of the controller way easier. Also, the lack of dynamic components involved in the model, makes the controller transferable between vehicle's which may not have the same dynamic characteristics.

There are different kinematic models that can be used to determine the model of a vehicle. Nonetheless, one of the most extended procedures is to use the **kinematic bicycle model**, a classic model which has been widely used due to its simplicity and good results. The model represents the vehicle as a system with two axles, with a determined distance between them (called *wheelbase*, the addition of the front and rear distances). The front wheel of the vehicle can be turned a certain angle (δ_f), and the heading of the vehicle (Ψ) is defined with respect to the global system of coordinates. The orientation of the vehicle, called **chassis slip angle** (β) is described by equation (3.5).

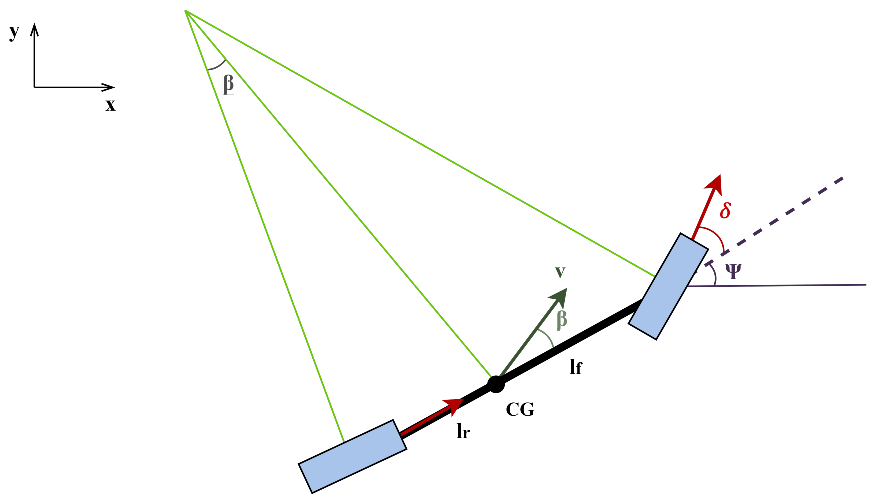


Figure 3.5: Bicycle kinematic model, reference point at the centre of gravity CG.

$$\beta(k) = \arctan(\frac{lf \cdot \tan(\delta_f(k))}{l_f + l_r}) \tag{3.5}$$

$$\tag{3.6}$$

It is important to note that, in this project, this model is **only used when the speed of the vehicle is very low**. Therefore, for this model it is considered that the lateral velocity of the vehicle can be neglected, meaning increments are only affected by x-components. Consequently, the kinematic model of the vehicle is described by the state equations (3.7).

$$\begin{aligned} x(k+1) &= x(k) + \dot{x}(k) \cdot \cos(\Psi(k)) \cdot dt \\ y(k+1) &= y(k) + \dot{x}(k) \cdot \sin(\Psi(k)) \cdot dt \\ \Psi(k+1) &= \Psi(k) + \frac{\dot{x}(k) \cdot dt \cdot \cos(\beta(k))}{l_f + l_r} \cdot \tan(\delta_f(k)) \\ v(k+1) &= v(k) + \ddot{x}(k) \cdot dt \end{aligned} \tag{3.7}$$

3.3.2 Dynamic Model

When working at high speeds, vehicles do not usually satisfy slip conditions: forces acting in the system cannot be neglected, since their effects affect significantly the motion of the vehicle. Consequently, dynamic modeling of the systems is also important. This model deals with the forces and torques that act and interact with the system. As the kinematic model used in this project is the bicycle model, the dynamic model developed has been based taking as starting point this bicycle model. As an assumption, the longitudinal velocity of the vehicle is considered constant; also, suspension, road inclination and any aerodynamic influences are considered null. The lateral force of the tires is considered to be a linear function, dependent on the slip angle. It is important to note that the latter is only valid to scenarios with **low lateral accelerations**.

Considering that the longitudinal speed, $\dot{x}$, is constant and strictly positive, Newton's second law (3.8) is applied.

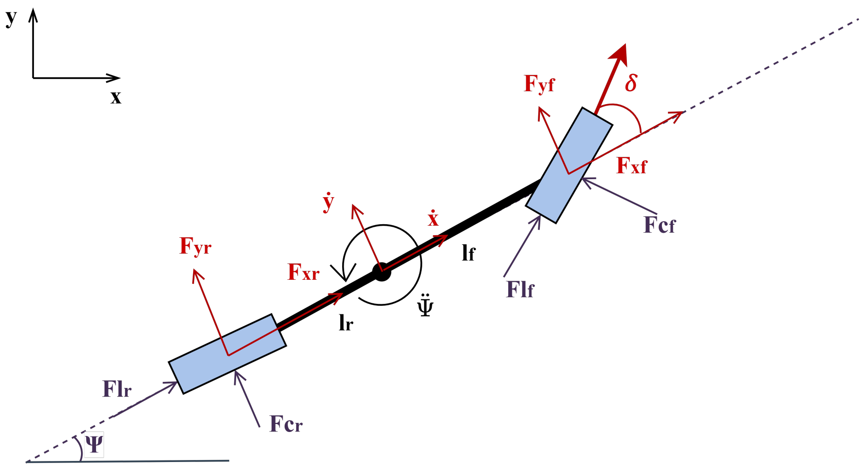


Figure 3.6: Bicycle dynamic modelling notation. Representation of the forces in the vehicle body-fixed frame (F_x, F_y), the forces in the tire-fixed frame (F_l, F_c) as well as the rotational and translational velocities. Source ([4])

$$m \cdot \ddot{y}(k) = F_f(k) + F_r(k) \tag{3.8}$$

F_f and F_r denote the lateral tyre forces of the front and rear wheels. Also, the balance of moments (3.9) can be calculated.

$$I_z \cdot \ddot{\Psi}(k) = F_f(k) \cdot l_f + F_r(k) \cdot l_r \tag{3.9}$$

With this, the yaw dynamics of the vehicle are defined. Additionally, θ_f (3.10) and θ_r (3.11) correspond to the angles between the velocity vector and the longitudinal speed direction $\dot{x}$. With this, the **chassis slip angle** (3.12) can be determined.

$$\theta_f(k) = \arctan(\frac{\dot{y}(k) + l_f \cdot \dot{\Psi}(k)}{\dot{x}(k)}) \tag{3.10}$$

$$\theta_r(k) = \arctan(\frac{\dot{y}(k) - l_r \cdot \dot{\Psi}(k)}{\dot{x}(k)}) \tag{3.11}$$

$$\beta(k) = \delta_f(k) - \theta_f(k) \tag{3.12}$$

The lateral tyre forces are proportional to the side-slip angles, if these are small. Consequently, the lateral forces can be described by the following equatuon 3.13.

$$\begin{aligned} F_f(k) &= 2 \cdot C_f \cdot (\delta_f(k) - \theta_f(k)) \\ F_r(k) &= 2 \cdot C_r \cdot (-\theta_r(k)) \end{aligned} \tag{3.13}$$

The vehicle's displacement follows an **uniform accelerated motion** behaviour. Therefore, the

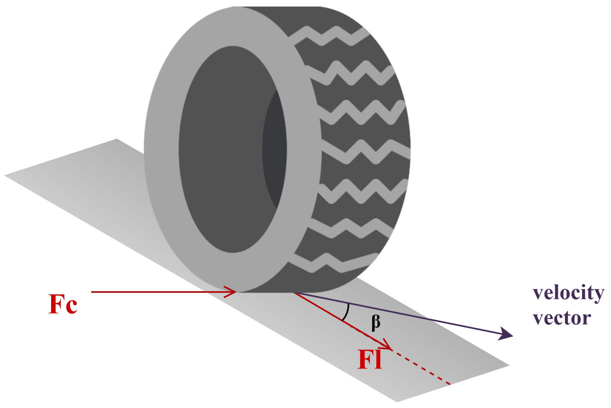


Figure 3.7: Representation of the forces in the tire-fixed frame (F_l, F_c).

travelled distance at each distance corresponds to equations (3.14).

$$\begin{aligned} d(k) &= \sqrt{x_{veh}(k)^2 + y_{veh}(k)^2} \\ x_{veh}(k) &= \frac{1}{2} \cdot \ddot{x}(k) \cdot dt^2 + \dot{x}(k) \cdot dt \\ y_{veh}(k) &= \frac{1}{2} \cdot \ddot{y}(k) \cdot dt^2 + \dot{y}(k) \cdot dt \end{aligned} \tag{3.14}$$

Also, the vehicle's acceleration (3.15) can be obtained from the equations previously formulated. For the evaluation of the variables, accelerations and vehicle's yaw rate are initially measured with vehicle's inertial sensors.

$$\begin{aligned} \ddot{y}(k) &= \frac{F_f(k) + F_r(k)}{m} \\ \ddot{\Psi}(k) &= \frac{l_f \cdot F_f(k) + l_r \cdot F_r(k)}{I_z} \end{aligned} \tag{3.15}$$

The previous equations estimate the translational and rotational dynamics of the vehicle. They are essential to evaluate kinematically the vehicle's position and orientation in the global system of reference.

Consequently, the dynamic model of the vehicle is described by the set of equations (3.16).

$$\begin{aligned} x(k+1) &= x(k) + d(k) \cdot \cos(\Psi(k)) \\ y(k+1) &= y(k) + d(k) \cdot \sin(\Psi(k)) \\ \Psi(k+1) &= \Psi(k) - \dot{\Psi}(k)dt \\ v(k+1) &= v(k) + \ddot{x}(k) \cdot dt \end{aligned} \tag{3.16}$$

In addition to the described models of the vehicle, the **actuators** of the own vehicle must also be considered in order to design the controller. Physical constraints affecting the actuators, like speed limitations and maximum steering wheel turns, restrict the controller solutions. The combination between both the kinematic and dynamic models and the physical constraints rule the MPC behavior. This is studied in subsection 3.4.2.

3.4 Control Action

Model Predictive Control relies upon the tracking of a reference trajectory. When applied to autonomous driving, the trajectory the vehicle follows is obtained by minimizing the cost function of the control process. Here, the term **cost function** refers to the error included in the process; that is to say, the difference between where the vehicle must go (*reference*) and where it is going to go (*prediction*). When working with a MPC, the controller generates several simulated inputs coming from the actuator (e.g. a certain signal for the steering wheel and the throttle). According to these input simulations, resulting predicted trajectories are obtained. The trajectory with the minimum cost, meaning the one with the minimum error, is selected; in every loop, a polynomial function optimizes the trajectory given by the controller and the model of the vehicle. The control action that generates this trajectory is the one finally implemented in the vehicle. After this, all the states are updated, and the process is repeated.

3.4.1 Cost Function

The term previously mentioned, known as **cost function**, refers to a performance index which needs to be minimized (or maximized) in order to perform accurately the system's control action. This function includes, in general terms, the error in the output signal (*state cost*) as well as the error present in the control action (*input smoothness cost*). There are different procedures for obtaining the optimal cost function. Nonetheless, the most frequent procedure is the so know **General Optimal Control cost function** [21].

$$J = x(t_f)^T Q_{x_1} x(t_f) + \int_{t_o}^{t_f} (x(k)^T Q_{x_2} x(k) + u(k)^T Q_u u(k))dt \tag{3.17}$$

Here, each Q-term corresponds to the weight of each element in the cost function: they denote the importance of the determined element in the optimization process. This specific cost function (3.17) is called **quadratic cost function**, since all the states are squared in order to reduce the complexity of the problem. Due to the matrix form of the elements, the quadratic form is denoted with transposed elements. The first element of the function is the *terminal cost*, whereas the integral part is called the *running cost*. The structure of the cost function determines the type of control problem working with.

In particular for this project, the **state cost** term implemented in the MPC considers possible errors in the four state parameters of the system: a heading error (Ψ_e), a velocity error (v_e), and a lateral error (cte). Additionally, an **input smoothness cost** is also included: variations in the acceleration (Δ_Θ) and the steering angle ($\Delta\delta_f$) are considered. As before, the Q terms correspond to the weight of each cost term. For simplification purposes, the prediction and control horizons are identical (N).

$$J = \sum_{i=1}^{Hp} e^T Q_e e + \sum_{i=0}^{Hc-1} \Delta u^T Q_u \Delta u$$

$$\begin{aligned} J = \sum_{i=1}^{N} \Psi_e(k+i)^T Q_\Psi \Psi_e(k+i) + \sum_{i=1}^{N} v_e(k+i)^T Q_v v_e(k+i) + \sum_{i=1}^{N} cte(k+i)^T Q_{cte} cte(k+i) + \\ + \sum_{i=0}^{N-1} \Delta\delta_f(k+i)^T Q_\delta \Delta\delta_f(k+i) + \sum_{i=0}^{N-1} \Delta\Theta(k+i)^T Q_\Theta \Delta\Theta(k+i) \end{aligned} \tag{3.18}$$

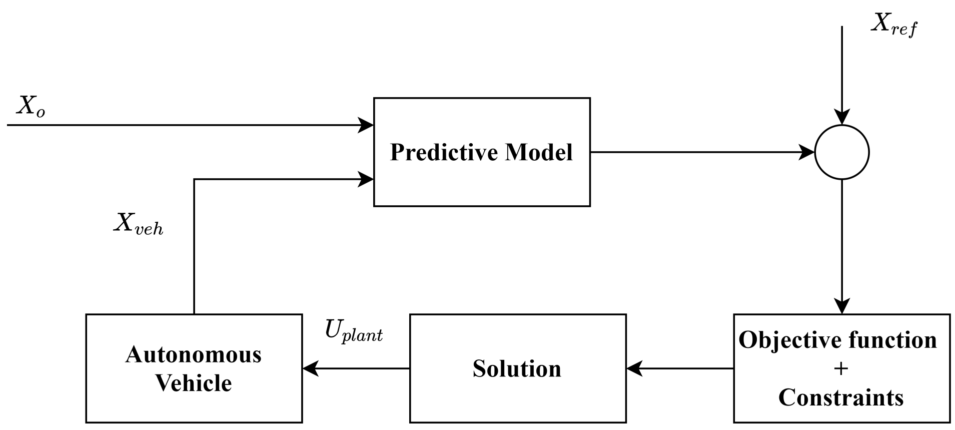


Figure 3.8: Controller structure diagram apllied to autonomous vehicles.

3.4.2 Control Action Constraints

Constraints for the control action terms are included inside the MPC, in order to prevent abrupt motion changes and ensure the working range of the actuators. In this way, the motion of the vehicle becomes smoother and more safe over time. Also, weights are different for each term in the cost function, in order to ease any possible future tuning. In particular, the constraints included in the autonomous vehicle problem studied are described in the following subsections.

3.4.2.1 Tire Behaviour

The safety of the manoeuvre is greatly affected by the lateral forces suffered by the tires. As a consequence, the dynamic characteristics of the plant are determinant for the system's behaviour. Lateral forces affecting the tires are inversely proportional to the vehicle's adherence to the road; the higher they are, the worse the adherence of the vehicle. The slip angle 3.12, is defined as the difference between the angle determined by the velocity vector and the wheels' steering angle. This relation implies the relation between high slip angles and the loss of control of the vehicle. Therefore, this work proposes the restriction of this parameter, in order to allow the controller to manoeuvre under safe conditions. The maximum slip angle determined for this work is $\beta_{max} = 2.5°$

3.4.2.2 Physical Limitations of the Actuators

As it has been previously mentioned, the control action of the system is composed of the steering wheel angle and the throttle's position, denoted in equation 3.19.

$$U_{veh} = [\delta_f, \Theta_t] \tag{3.19}$$

As actuators are mechanical components, they are physically constrained. They have ranges of operation, and outside of them they do not guarantee their working characteristics. As a consequence, constraints are added in the MPC in order to consider these mechanical limitations. These are denoted in equation 3.21.

$$-540° \leq \{\delta_f\} \leq 540° \tag{3.20}$$
$$0 \leq \{\Theta_t\} \leq 1 \tag{3.21}$$

It is important to note that, additionally, the possible increment of these variables is also constrained. This is done with the purpose of obtaining slow changes in between the different states; driving becomes smoother, and therefore more comfortable for the driver.

3.4.2.3 Time Constraints

On the other hand, Model Predictive Controllers work in a **discrete time domain**: they predict the states in specific future instants of time. These instants of time, as well as the time between them, are the so-called **sample time** and **prediction horizon**. When tuned in, these controller parameters determine the accuracy of the results.

- **Sample Time (T_s).** It denotes the rate of execution of the control algorithm. The higher the sample rate, the worse the response to disturbances. The lower the sample rate, the better the response to disturbances (but with an excessive computational effort).
- **Horizon (N).** For each time step, the horizon decides how many predictions and future control signals have to be considered. Normally, two horizons are used: the **prediction horizon** (H_p) and the **control horizon**(H_c). With regard to the control horizon, it refers to the time steps determined for the process. As it directly relates to the computation of the algorithm, the complexity of the process increases with the control horizon. Usually, H_c is smaller than H_p in order to reduce the optimization problem. However, as in this work, it is also frequent to find that both horizons are equal in order to simplify the design.

When combined together, constraints and time parameters condition the control action of the MPC. These terms conform the range of action of the process, by determining safety limits and operating conditions, as well as the velocity of execution and the response the controller can guarantee.

Chapter 4

Obstacle Avoidance Control

4.1 General Aspects

Some specialists in the field consider that vehicle planners have multiple layers. As it has been previously mentioned, S. C. Pendleton [9] denotes that planning modules have generally a three-level structure, consisting of a mission planning module, a behavioral planning module and a motion planning module. The most general (based on goals) are the mission planning modules, whereas behavioral and motion planning layers consist of more complex planning structures. Behavioral modules are used when dynamic obstacles are not considered inside the environment, thus only an anticipation of seconds or minutes can be done in order to avoid any possible collision. However, motion modules are the most reactive; they are sometimes called **hazard avoidance modules**, since they are the fastest and most reactive in case of detecting a possible collision with an object in the environment.

As concluded by S. C. Peters[22], the **hazard avoidance capability** of these modules can be introduced in any navigation system through the proper use of sensor, motion and control systems. With the proper information gathered from the environment, determinant characteristics of hazards can be used in order to avoid them. However, motion planning and control when hazard avoidance is introduced is challenging due to several reasons, primarily trajectory constraints derived from the vehicle physical and stability limits. Also, computation time is a critical parameter which influences the results of the process: the length of prediction horizons can determine the consideration of the hazards. Long prediction horizons may improve performance, but with an increasing computational demand; also, short prediction horizons can disregard important information for the control analysis.

Commonly, **obstacle avoidance** is not treated as a control problem, but as an independent difficulty solved by following a planning strategy: if a determined position in the environment should be avoided, the path is recalculated. However, this avoidance problem formulation can alternatively be seen as a control problem. Avoidance can be used to predict the control of the system, and integrate the needed constraints for the states to solve the problem only with the proper control actions. However, for doing so, a new term must be taken into consideration. Now, the procedure of moving from an initial point A to a goal point B only by knowing a set of points in the space domain, changes. When a certain obstacle is considered inside the environment, the autonomous vehicle must avoid the collision with this object and consider its overtake, while following the reference trajectory. For this, an **obstacle avoidance term** must be implemented, since the obstacle is not included in the original trajectory tracking control problem. According to V. Turri et al [23], an effective way to perform this obstacle avoidance relies upon the introduction of additional safety constraints for the lateral error. It is supposed that the actual and future positions of the object are known, as well as that the controller knows the side of the obstacle which is safe to pass. These two assumptions define a safe region through which the vehicle shall move.

In the same way, A. Franco et al [5] propose an optimization problem based on the **cost minimization under constraints**. This proposal suggests the definition of an area through which the vehicle must not enter during the prediction horizon. The redefinition of this area is done in every control interval, by continually updating 5 constraints: left and right boundaries of the road, a constraint for obstacle avoidance, a position of the closest obstacle and the vehicle's position. With this, the vehicle is maintained inside a specific lane, while the possible obstacles located on it are avoided. This algorithm is very flexible, which allows its application in multiple scenarios.

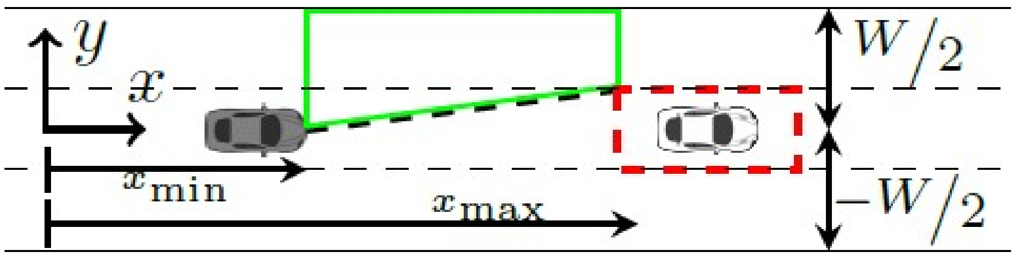


Figure 4.1: Constraints in the left overtaking scenario. Source [5].

Similarly, F. Lin et al [24] approach the obstacle avoidance problem by introducing a **penalty function** 4.1, in order to readjust the cost function to the deviation between the obstacle and the reference trajectory.

$$J_{obs} = Q_{obs} \frac{\dot{x}^2 + \dot{y}^2}{(x_{obs}^2 + x_{veh}^2) + (y_{obs}^2 + y_{veh}^2) + \Delta} \tag{4.1}$$

By introducing this J_{obs} penalty in the general cost function a collision avoidance is obtained, while the deviation from the initial reference is minimized. The closer the vehicle is to the obstacle, the bigger the cost term will be, and therefore, the bigger its importance in the optimization problem.

On the other hand, C. Hu et al[25] propose a different alternative in order to solve the obstacle avoidance problem in autonomous driving. Their control strategy differentiates between static and dynamic obstacles. The obstacle analysis conditions the consequent path planning and tracking. In the static obstacle scenario, a safe distance between the obstacle and the vehicle is considered. This parameter is included in the cost function as an additional penalty, like in F. Lin project development.

4.2 Obstacle Avoidance using safety constraints

In this project, the strategy applied does not combine only one of the methods mentioned, but a combination of them. The approach of this project can be compared with the **elastic bands method**, developed by S. Quinlan et al [6]. Elastic bands are defined as a *deformable collision-free path*, with an initial shape given by a planner. These bands may deform their path in real time, according to any possible obstacle or change present in the environment. In this way, the system can accommodate uncertainties and react to any unforeseen circumsntances. Elastic bands connect the system and the environment, while preserving the global nature of the original planned path. The strategy followed when applying elastic bands has three levels:

1. **Level 1 - Path planning.** Generation of the original path wit a predefined model.
2. **Level 2 - Elastic Bands.** Deformation of the original path in real time, according to the changes perceived in the environment.
3. **Level 3 - Control.** The vehicle is moved along the elastic band.

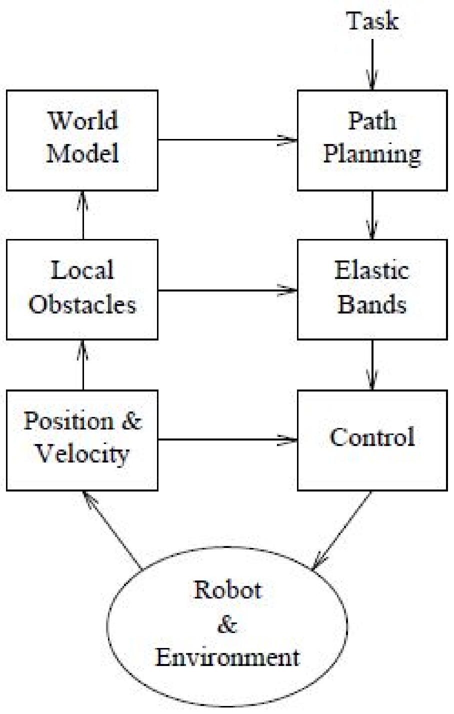


Figure 4.2: Three-level hierarchy for a system executing a motion task. Source [6].

By applying this procedure, it is unnecessary to recall the path planner every time there is an unexpected change in the environment, since the system can react in real time with the information gathered from the environment by sensors. In his research, S. Quinlan denotes that elastic bands are represented by a finite number of bubbles, which are obtained from the possible configurations and points of the system. These overlapping bubbles ensure that whenever the vehicle is inside them, the path will be collision free. The size and shape of the bubbles is as complex as the situation encountered: the further the obstacle, the bigger the bubble. That is to say: the further the obstacle, the bigger the collision free region.

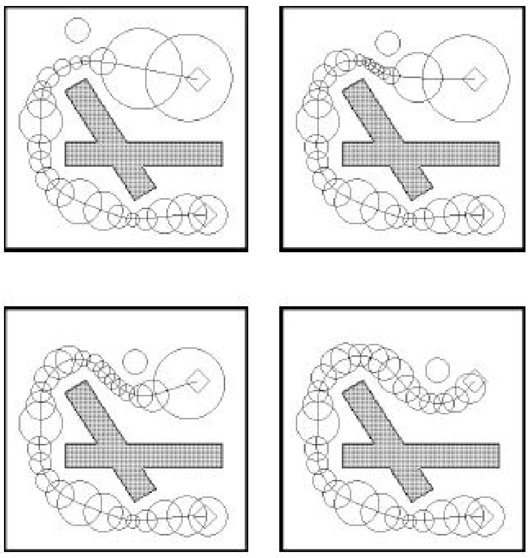


Figure 4.3: Deformation of the elastic band according to the position of the obstacle. Source [6].

The procedure followed by S. Quinlan to deform the bubbles is ruled by **artificial contraction and repulsion forces**, which reduce any conflictive space and introduce clearance around the path. However, the deformation of the bands applied in the MPC developed for this project is not ruled by forces, but by constraints. Supposing there is a static obstacle in the environment, the way to deform the original elastic band of the vehicle is by introducing a series of constraints inside of the MPC. In this way, a possible collision of the vehicle is avoided while following the reference trajectory. In this project, the path planning and elastic bands levels are somehow merged into just one. The development of the control action that is finally implemented to move the vehicle, considers obstacle constraints inside the optimization process.

4.2.1 Description of the Algorithm

Given that the vehicle follows a given trajectory, it is supposed that several obstacles are included inside the environment. In order to avoid the collision between the autonomous vehicle and the obstacles, diverse distance constraints are considered. Whenever these constraints are not fullfilled by the control action, this one will be recalculated. In other words, the optimization process does not finish until both the lateral error and the obstacle avoidance problem are solved. With this, the vehicle follows the desired reference trajectory while ensuring that it will neither collide with the obstacle nor overreact and separate innecesarily from the reference.

The procedure developed in the optimization process is based on the use of **homogeneous transformation matrices**. These matrices relate the different coordinate systems: the general, the vehicle and the obstacle. In this way, the relative position of the obstacle with respect to the vehicle can be known.

$$T_o^{obs} = T_o^{veh} T_{veh}^{obs}$$

Given that the vehicle and obstacles position and orientation are known, the resulting matrix is the

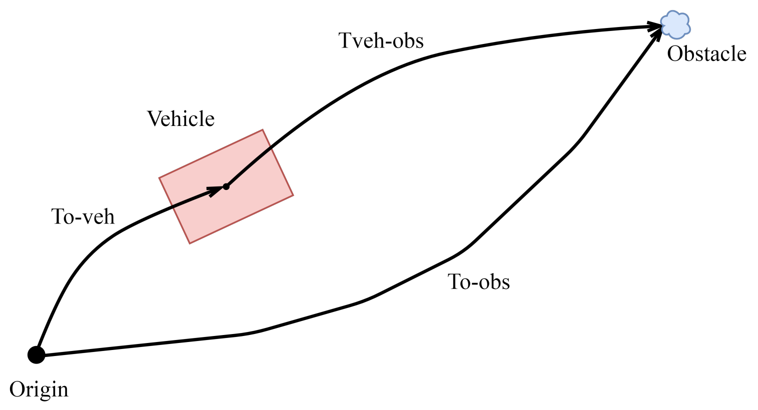


Figure 4.4: Transformation graph for the matrices associated with the vehicle and the obstacle.

following 4.2[1].

$$T_{veh}^{obs} = \begin{pmatrix} R_{3x3} & p_{3x1} \\ 0 & 1 \end{pmatrix}$$

$$T_{veh}^{obs}(k) = \begin{pmatrix} \cos(\Psi(k)) & -\sin(\Psi(k)) & 0 & \cos(\Psi(k))(x_{obs}(k) - x_{veh}(k)) + \sin(\Psi(k))(y_{obs}(k) - y_{veh}(k)) \\ \sin(\Psi(k)) & \cos(\Psi(k)) & 0 & -\sin(\Psi(k))(x_{obs}(k) - x_{veh}(k)) + \cos(\Psi(k))(y_{obs}(k) - y_{veh}(k)) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.2}$$

By knowing the translational vector for the obstacle (p_{veh}^{obs}), it is possible to determine the **longitudinal distance** between the vehicle and the obstacle. In case of a multiple obstacle scenario, the longitudinal distance between the vehicle and the obstacle is needed, in order to know the closest collision-problem to the vehicle. This distance is the x-component of the translational vector p_{veh}^{obs}, defined by equation (4.3).

$$d_{lon}(k) = \cos(\Psi(k))(x_{obs}(k) - x_{veh}(k)) + \sin(\Psi(k))(y_{obs}(k) - y_{veh}(k)) \tag{4.3}$$

The obstacle with the shortest longitudinal distance ($min(d_{lon}(k))$) is chosen in order to develop the optimization of the control action. This longitudinal distance will be referred as the **detection distance to the obstacle**.

Moreover, the **lateral distance** to the obstacle must be considered in order to solve the avoidance problem. By knowing it, an additional safety restriction can be determined. This lateral distance results from calculating the distance error between the reference trajectory and the vehicle's position at instant k, denoted at equation (4.4). This is valid supposing that **the obstacle is positioned within the reference trajectory**. In this way, lateral distance constraints can be included; with them, the minimum and maximum distance between the vehicle and the obstacle can be specified.

$$d_{lat}(k) = x_{reference}(k) - x_{veh}(k) \tag{4.4}$$

s

It is important to note that an additional parameter, called **slope avoidance term**, is also included. This term represents the slope at which the vehicle ought to perform the collision avoidance manouevre.

[1]Note that the z coordinate is null.

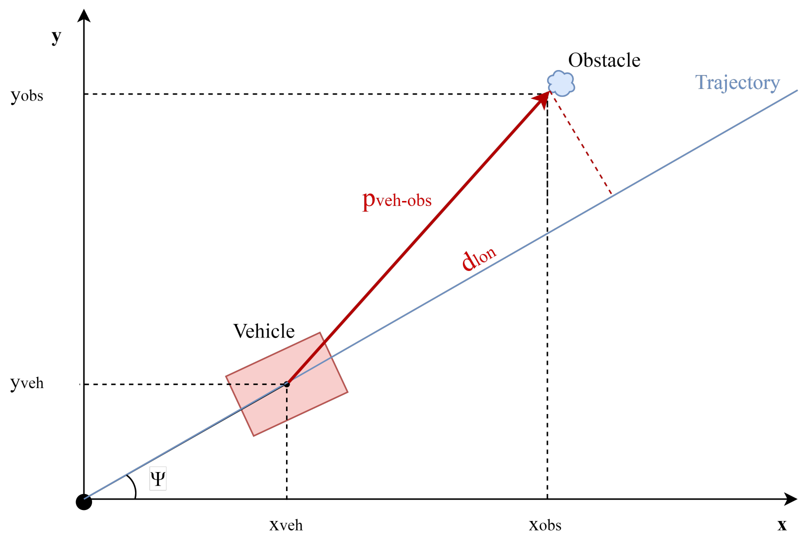


Figure 4.5: Translational vector of the obstacle, related to the vehicle system of reference.

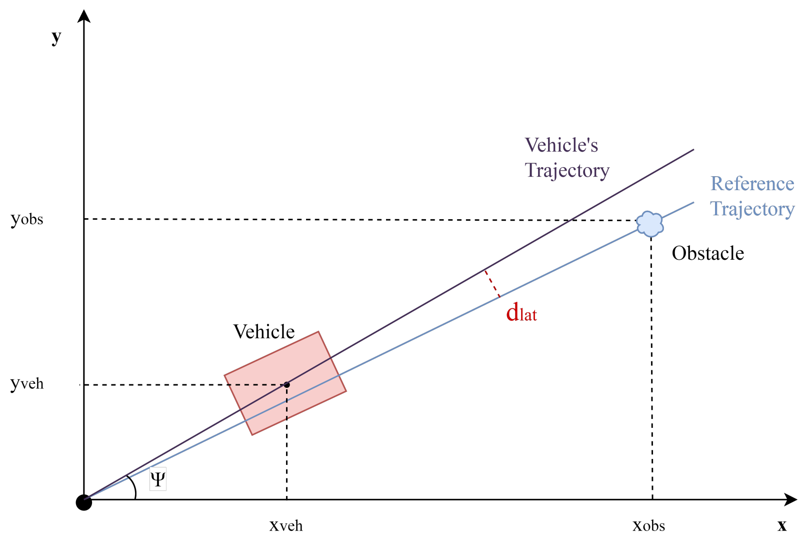


Figure 4.6: Translational vector of the obstacle, related to the vehicle system of reference.

This term implies the smoothness of the manoeuvre, and avoids abrupt changes in the trajectory of the obstacle.

4.2.2 Obstacle Avoidance Constraints

As in section 3.4, additional constraints are needed in order to ensure the safety of the action taken by the controller.

4.2.2.1 Morphology of the road

In order to perform an obstacle avoidance manoeuvre, the morphological characteristics of the road are needed. For instance, the road width can determine if a manoeuvre for avoiding collision should be executed through the left or right side of the obstacle. In this case, the vehicle's distance to the lane limits is given by the simulator. This is used to evaluate the distance to the lane limits; the centre of the vehicle ought to be $2m$ apart from the latter, as depicted in equation 4.5.

$$\{lane_{limit}\} \leq 2m \tag{4.5}$$

4.2.2.2 Lateral distance to the obstacle

As mentioned in subsection 4.2.1, there must be a minimum lateral distance between the vehicle and the obstacle to avoid collision. This cosntraint is set to $3m$, with a tolerance of $\pm 0.1m$. The MPC is responsible for providing a control action, either overpassing throught the right or the left, but always fulfilling the mentioned constraint. Equation 4.6 shows the obstacle lateral distance constraint.

$$\{lateral_{distance}\} \leq 3m \tag{4.6}$$

Chapter 5

Results

5.1 Experimental background

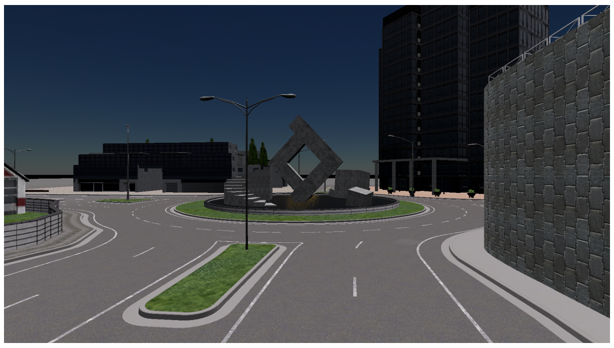
5.1.1 CARLA Simulator

When working with autonomous vehicles, the training and testing stage becomes more complicated than in other processes. The difficulties and impediments derived from logistical and infrastructure costs provoke that this stage cannot be performed in real systems. An alternative to this real testing is the use of simulators to train and validate the strategies developed, especially the most innovative ones. Over time, the autonomous driving simulators have become essential tools for this field.

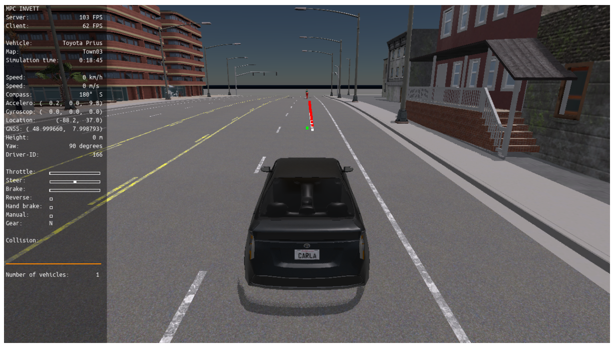
In this project, the CARLA open-source autonomous driving simulator has been remarkably useful for the validation of the controller. This open source simulator has been developed by Intel Lab, Toyota Research Institute and the Computer Vision Center in Barcelona, and is under constant development due to its large community of users and researchers. Its main objective is to *support training, prototyping, and validation of autonomous driving models, including both perception and control* [26] . It has been built on a C++ engine, and can be controlled by an external client script. The vehicle can be controlled manually (with a graphical user interface, implemented in a Python API module). Indeed, the vehicle can be automated, with the addition of sensors that introduce provide information about the environment and can be determinant for the decisions taken along the process.

CARLA simulator is based on a server-client architecture: the server is responsible for running the simulation, whereas the client API interacts between the autonomous agent and the server. The commands interchanged control the vehicle, but also are used to control the server's behaviour. In this way, the simulator is able to provide a dynamic world with a simple interface. CARLA contains a wide variety of static and dynamic environment alternatives: vehicle models, buildings, traffic signs, pedestriants, urban scenarios, etc. Also, environmental conditions and illumination can be modified.

One of the main disadvantages of CARLA simulator is that it is a performance demanding software. The developers of the simulator reccomend to have at least 30-50GB free in order to build the program, as well as at the very minimum 4GB GPU [27]. If not, the computer will not be powerful enough, and it will not be able to provide reliable testing results. As a consequence, this work has been entirely tested by launching the simulator with low quality. This launching disables all post-processing and shadows, and the environment drawing is done 50m apart, and not infinite. With this, the simulator is able to run faster, and with no consequences in the testing. This work has run the simulator on the graphic card Titan RTX NVIDIA, while the MPC controller runs on an Intel i7 core at 2.9GHz.



(a)



```
MPC INVETT
Server:              103 FPS
Client:               62 FPS

Vehicle:         Toyota Prius
Map:                   Town03
Simulation time:      0:18:45

Speed:                 0 km/h
Speed:                  0 m/s
Compass:              180°  S
Accelero: (  0.2,  0.0,  9.8)
Gyroscop: (  0.0,  0.0,  0.0)
Location:       (-88.2, 37.0)
GNSS: ( 48.999660,  7.998793)
Height:                   0 m
Yaw:               90 degrees
Driver-ID:                166

Throttle:
Steer:
Brake:
Reverse:     □
Hand brake:  □
Manual:      □
Gear:        N

Collision:

Number of vehicles:        1
```

(b)

Figure 5.1: CARLA simulator environment. (a) General CARLA environment. (b) TOYOTA Prius model, used for testing the MPC controller.

```
$ ./CarlaUE4.sh -quality-level=Low
```

5.1.2 Strategy and methodology

In order to make the most out of the implemented controller, a proper tuning of the latter ought to be done before testing. With this, the optimal results for each test will be obtained. This tuning procedure is done by modifying different parameters of the controller. However, it can also involve the modification of different characteristics of the plant model, depending on the objective of the test.

By default, the configuration of the modifiable parameters is denoted in section A. The tables included correspond to the kinematic and dynamic models (section A.1), the controller (section A.2) and the obstacle avoidance problem (section A.3).

5.1.2.1 COBYLA Optimization Algorithm

The optimization algorithm used to minimize the cost function is the COBYLA algorithm: a Constrained Optimization BY Linear Approximation. It is a a gradient-free optimization algorithm capable of handling nonlinear inequality constraints. The convergence of COBYLA is slow; however, its high stability and the few parameters needed to tune it make it very popular. This algorithm is used to solve the optimization problem denoted in equation (3.18).

In general terms, COBYLA is an optimization method implemented when the derivative of the objectie function is unkown (gradient-free optimization). In mathematical terms, it can find the vector $\vec{x} \epsilon S, with S \subseteq \mathbb{R}^n$ that has the minimum or maximum value $f(\vec{x})$, with no need of knowing the gradient of f [28]. The implementaiton of this algorithm relies on the iterative approximation of the constrained problem. Every iteration evaluates a new data point in the space, in order to improve the approximation for the following iteration. When the improvement finishes, the step size is reduced until it refines the results.

COBYLA algorithm works by iteratively approximating the actual constrained optimization problem with linear programming problems. During an iteration, an approximating linear programming problem is solved to obtain a candidate for the optimal solution. The candidate solution is evaluated using the original objective and constraint functions, yielding a new data point in the optimization space. This information is used to improve the approximating linear programming problem used for the next iteration of the algorithm. When the solution cannot be improved anymore, the step size is reduced, refining the search. When the step size becomes sufficiently small, the algorithm finishes.

The selection of this algorithm has been done by considering the good results it obtains when working with convex optimization problems, like the cost function implemented in this MPC.

5.1.2.2 Time Parameters

Generally speaking, iterations correspond to the systematic application of a function, where the output of an iteration is used as the input for the next. This repetitive procedure delivers the optimized results. However, the number of iterations is dependent on the computer characteristics and power limitations: an excessive increase of iterations would imply the slow execution of the complete process. As a consequence, the maximum number of iterations of the optimization algorithm in this work is set to 200. In the determined sample time T_s, it would need to execute more times the algorithm. This would impact in the final results of the simulation.

Moreover, as it has been mentioned in section 4.2.2.1, the control and prediction horizons of the controller have been made identical. Each term corresponds to a different concept: control horizon denotes the samples wanted from the model for each instant, whereas prediction horizon denotes the data from the reference wanted at each instant of time. However, in order to simplify the design process, they have been made identical: $H_c = H_p = N = 6$. This term has been chosen, as the previous parameters, according to the computational capabilities available: it has been considered the most appropiate value for this work, providing enough reliable information for the control of the vehicle. Since the time step defined in the plant model is variable, the control horizon will depend on the value of the latter at each instant.

It is important to emphasize the fact that, in the case of the obstacle avoidance algorithm, it is not based on all the time steps of the receding horizon defined. The maximum distance calculation is only considered in the last time step of the receding horizon ($N = 6$), as well as the minimum distance constraint. That is to say, when the optimization process is done, only the last instant considers a range of values for the lateral distance. If the optimization considered every constraint for each instant of the horizon, the solution obtained would be, theoretically, more precise. Nonetheless, it would also involve a bigger demand of power, in terms of computation, which influences the final result of the controller. Due to power limitations and the almost negligible differences between considering and neglecting them, only the last time step (N) considers the distance constraints. By doing this, the controller has more flexibility to reach the control objective. Nonetheless, the rest of the constraints mentioned are implemented in the complete horizon.

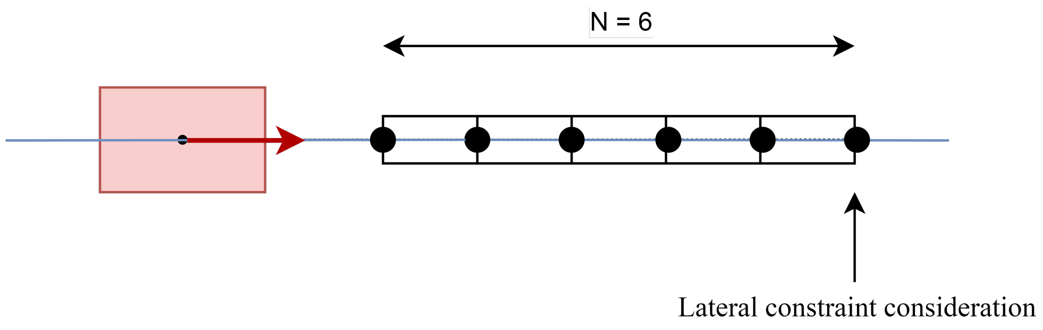


Figure 5.2: Consideration of the lateral distance constraint in the receding horizon, supposing N=6.

5.1.2.3 Weights of the Cost Function

The weights implemented in the cost function of the process (5.1) aim to reward those control actions of the system which make the vehicle's manoeuvres as smooth as possible, which is specially favorable for the obstacle avoidance problem. As a consequence, the weights which have the highest values correspond to the ones associated with the increments in the steering angle (Q_δ) and the acceleration (Q_Θ). The weight corresponding to the velocity receives a very low value, due to the disinterest of the velocity in the obstacle avoidance manoeuvre.

Table 5.1: Cost function weights for the control action.

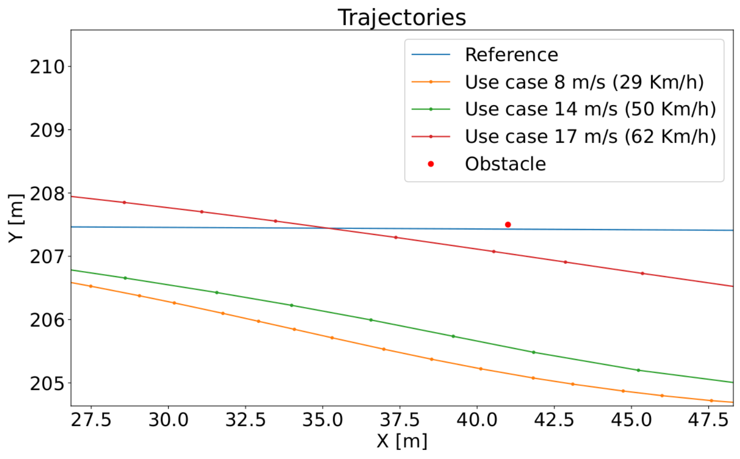
Parameter	**Description**	**Value**
Q_Ψ	Yaw weight	60
Q_v	Velocity weight	10^{11}
Q_{cte}	Cross-track error weight	$20 \cdot 10^{11}$
Q_δ	Steering angle increment weight	$100 \cdot 10^{11}$
Q_Θ	Acceleration increment weight	10^{11}

From this table, it can be seen that the initial weights of the cost function reward low steering angles and lateral errors (specially interesting in the obstacle avoidance problem).

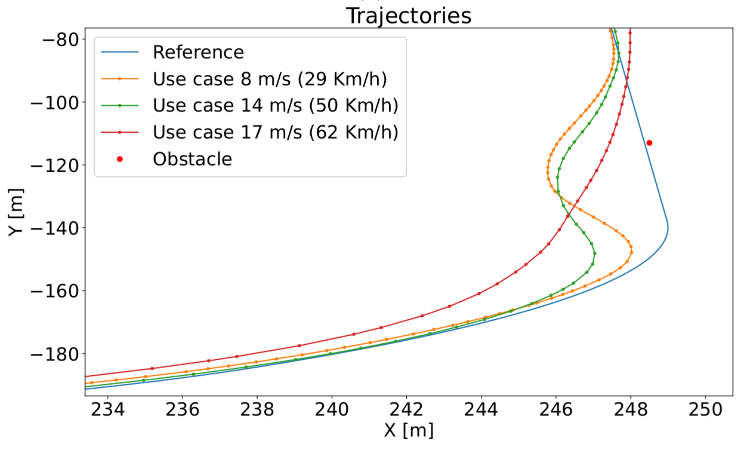
5.2 Experimental results

In the following pages, the study of the system response in two different situations is displayed. Each case differs in the position of the obstacle with respect to the road: a straight running lane, and before the entrance to a curved trajectory. The right lane is free in both cases, allowing the avoidance manoeuvre. CARLA integrates inertial sensors, needed by the MPC to measure the acceleration and angular velocity. Each case is used to test the reliability of the controller under different circumstances. The graphs displayed show the vehicle's trajectory as the obstacle is avoided, as well as the corresponding forces and lateral error obtained during the manoeuvres.

5.2.0.1 System Response



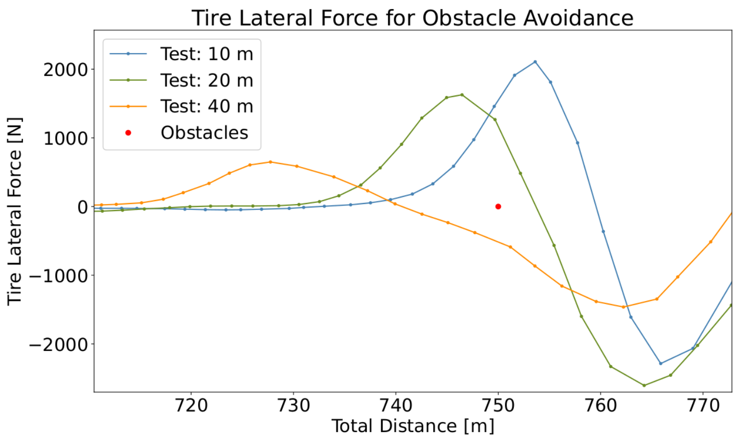
(a)



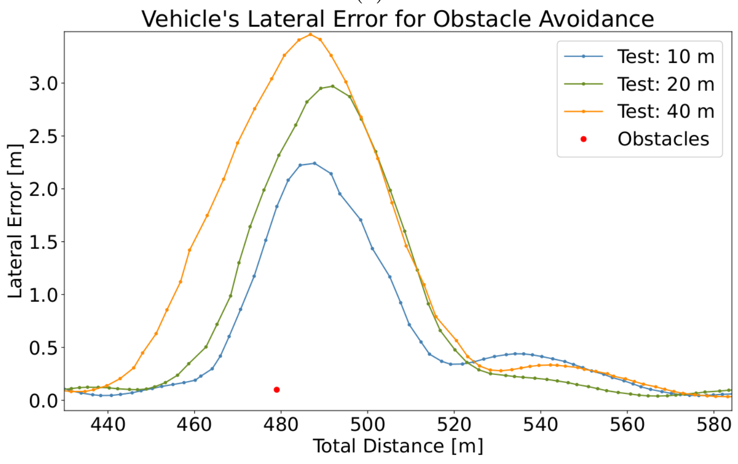
(b)

Figure 5.3: Cases of study results. (a) Straight trajectory. (b) Curved trajectory.

The graphs depicted before correspond to the simulation of the system at different speeds: $8, 14$ and $17m/s$ ($29, 50$ and $62km/h$, respectively). The static obstacle detection distance specified for the straight line case is of $40m$, for each mentioned velocity. The action taking place in the curved case determines the system response when, moving at $14m/s$, the detection occurs at the three different distances $(10, 20$ and $40m)$. All these detection distances distances are higher than the emergency braking distance of the vehicle. It is important to note that results are represented at the vehicle's centre of mass, positioning the obstacle in the middle of the lane. All the manoeuvres overpass the obstacle smoothly and safely, both for the driver and in this case, the pedestrian. Figure 5.3a depicts the positioning of the obstacle in a straight running lane, whereas figure 5.3 shows the results obtained when the obstacle is before entering a curved path.



(a)



(b)

Figure 5.4: Cases of study results. (a) Lateral forces affecting the tire. (b) Lateral distance to the obstacle.

On the other hand, the study of the lateral forces affecting the tires is also depicted 5.4a. These results correspond to a simulation where the vehicle, moving at $17 m/s$, detects the sat the three possible distances previously mentioned. The lateral tire force is inversely proportional to the detection distance. The explanation for the latter is based on the aggressivity of the manoeuvre which has to take place: the faster the system has to react, the higher the force the tire suffers. AS a consequence, the maximum lateral tire force $(2,000N)$ is given with the lowest detection distance, $10m$. From the dynamic model of the plant, it is known that the lateral force affecting the front tire is directly proportional to the vehicle's slip angle 3.13. This causes that, as seen in the simulations, the greatest turn of the wheel occurs when the reaction is needed fastly.

Figure 5.4b reflects the lateral error values obtained from these tests. This error increases simultaneously with the detection distance; this occurs due to the need of satisfying the constraints set for the controller. When the detection distance is low, the controller is not able to provide a control action which avoids the obstacle safely and, at the same time, fulfills all the constraints. This is mentioned in section 5.2.1. Nonetheless, even if the constraints are not completely fulfilled, the system always rewards those manoeuvres which are safe enough to be executed. That is to say, the vehicle will always be under control.

All the cases of the study show that the vehicle does not return slowly and smoothly to the original reference. This could be modified, but for safety reasons, the vehicle returns fastly to the reference trajectory after this safety manoeuvre has finished.

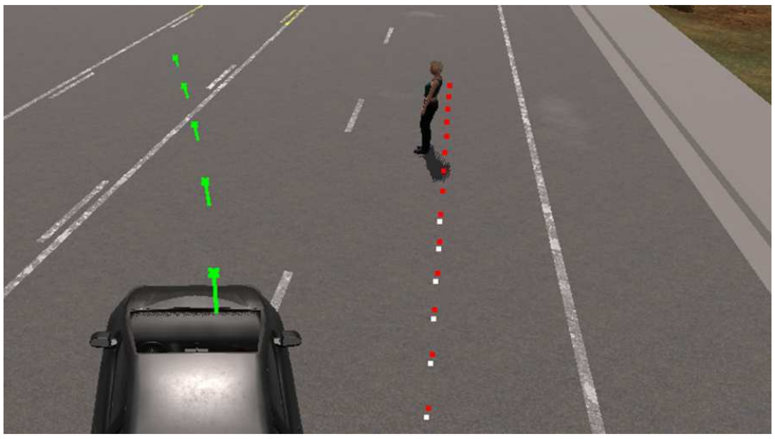


Figure 5.5: CARLA obstacle avoidance manoeuvre.

5.2.1 Statistical Analysis of the Lateral Error

A statistical analysis of the lateral error in the obstacle avoidance scenario is presented. The previous sections show that the controller is able to perform a collision-free trajectory, when there is an osbtacle on the road. Additionally, it has been shown that, when an obstacle avoidance manoeuvre occurs, it directly implies the impementation of a trajectory which differs from the initial reference. That is to say, an increase of the lateral error between the real trajectory and the reference will occur. In simulation tests, the lateral error obtained when the reference trajectory is strictly followed is in the range of $0.01 - 0.1m$. Nonetheless, when osbtacles are detected, this value increases significantly. Figure 5.7 shows the analysis of several cases of study, where several obstacles are placed along the vehicle's total trajectory. Each test is performed with different detection distances- 10, 20 and $40m$-, as well as at different throttle increments $(0.15, 0.25$ and $0.4m/s^2)$.

Firstly, a general approach cosidering the complete trajectory of the vehicle is presented.

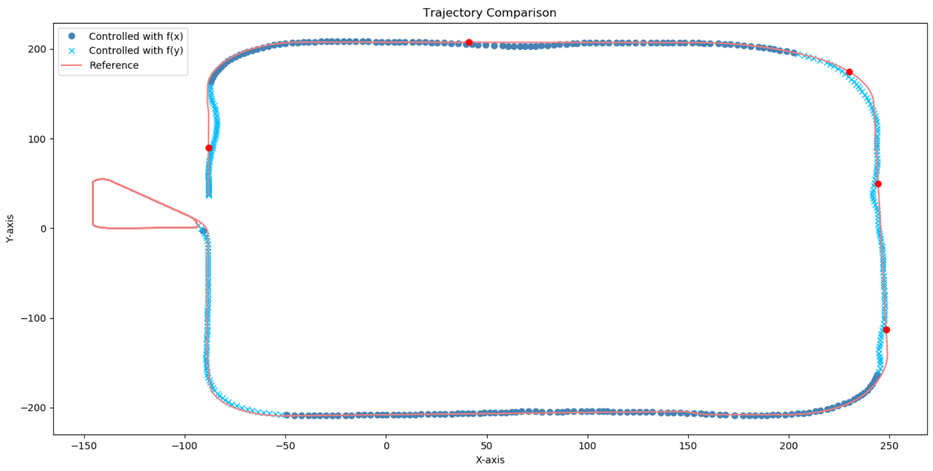


Figure 5.6: Vehicle's trajectory and obstacles' position for the statistical analysis. Test executed with a detection distance of $40m$, and a throttle increment of $0.4m/s^2$.

Table 5.2 denotes the general lateral error parameters obtained for the complete vehicle's trajectory of figure 5.6. The table shows how, in general terms, the lateral error commited is somehow constant and repetitive during the different testings.

Table 5.2: Average and Standard Deviation values for the different tests.

Detection	Θ increment	Average total error	Total Deviation	e_{max}	e_{min}
10 m	0.15	0.6973 m	0.7986 m	3.1144 m	0.0002 m
10 m	0.25	0.7731 m	0.8047 m	3.2154 m	0.0009 m
10 m	0.40	0.7734 m	0.8232 m	3.6292 m	0.0002 m
20 m	0.15	0.8183 m	1.0159 m	4.2012 m	0.0001 m
20 m	0.25	0.7877 m	0.9270 m	3.7986 m	0.0001 m
20 m	0.40	0.9472 m	0.9349 m	3.6635 m	0.0023 m
40 m	0.15	1.9502 m	1.3759 m	4.7832 m	0.0001 m
40 m	0.25	1.0787 m	1.2442 m	4.7398 m	0.0014 m
40 m	0.40	1.0335 m	1.1426 m	4.5015 m	0.0003 m

From these results, it is noticeable to see that in general terms, the lateral error is in the range of $0.7 - 2m$, respectively. That is to say, the error commited while the vehicle is following the reference trajectory is low enough to consider the approval of the system. Also, it is important to note that when the complete trajectory is considered, the deviation of the results increases as the lateral distance raises. This occurs since, whenever an osbtacle has to be overtaken with these conditions, the lateral distance of the vehicle reaches a maximum error value. This value is achieved due to the capability of the system to perform a manoeuvre safe enough to fulfill all the required constraints, including the lateral distance of $3m$.

Furthermore, table 5.3 gathers the concepts previously mentioned in the range of the obstacle lateral error. It can be seen that the highest lateral errors are found when the maximum detection distance is used. The maximum average lateral error, $4.2437m$, fulfills the lateral distance constraint and also implies a low deviation. This means that the difference between the errors obtained for each obstacle is minimum; therefore, the tuning of the parameters is satisfactory. On the other hand, results obtained for the lowest lateral error, $2.1516m$, certify that low detection distances and small acceleration increments worsen the lateral results and constraint requirements.

In the tests where a detection distance of $10m$ was used (tests a.1, a.2, a.3, purple boxes), the average maximum lateral error obtained is lower than the $3m$ distance constraint imposed (section 4.2.2.2). However, as this distance increases, the constraint begins to be fulfilled. This denotes that a bigger reaction distance for the vehicle to perform the manoeuvre conditions the safety of the controller. If the system has a large space to handle the collision-free aspiration, the results will finally be more satisfactory. Also, big increments in the acceleration condition this lateral error. The lowest deviations are found when velocity, at high speed, has the biggest increment ($0.4m/s^2$). The system will not execute a start-stop manoeuvre, due to the lack of time in order to perform this. Therefore, it will directly execute the action as safely as possible, even if the speed is high enough.

Figure 5.7 depicts how important the obstacle detection distance is for an autonomous vehicle. Since the obstacle manoeuvre has been considered, all the depicted lateral errors are not null. As the distance of detection increases, the maximum lateral error becomes higher, reaching its maximum value when the vehicle and the obstacle are side by side.

Table 5.3: Average and Standard Deviation values for the different tests.

Detection	Θ increment	Average lateral error	Standard Deviation
10 m	0.15	+2.1516 m	1.1458 m
10 m	0.25	+2.4780 m	0.4592 m
10 m	0.40	+2.8673 m	0.3819 m
20 m	0.15	+3.4633 m	0.6536 m
20 m	0.25	+3.1914 m	0.5282 m
20 m	0.40	+3.0009 m	0.7453 m
40 m	0.15	+4.1264 m	0.4001 m
40 m	0.25	+4.2516 m	0.4505 m
40 m	0.40	+4.2437 m	0.1985 m

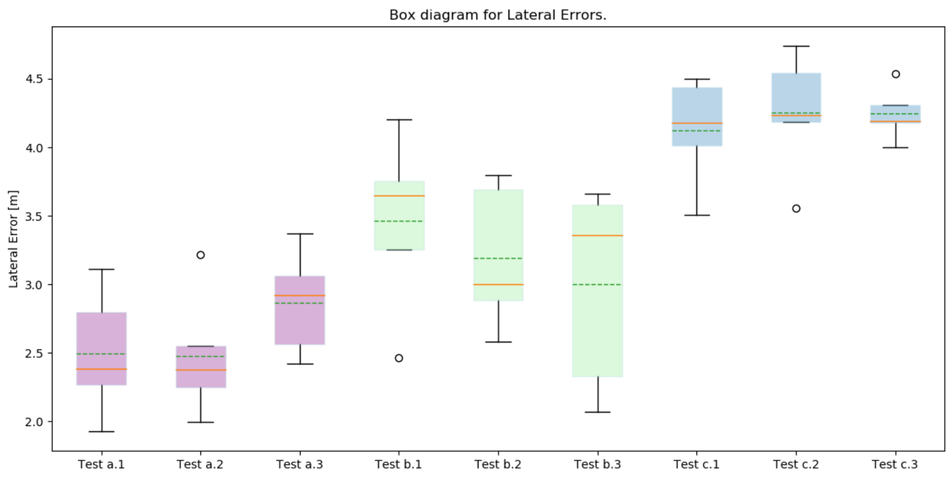


Figure 5.7: Box diagram for the lateral error in different tests. Average values are represented with green lines; median values are represented with orange lines. Letters a, b and c denote the detection distance $(10, 20$ and $40m)$; numbers 1, 2 and 3 correspond to the throttle's increment $(0.15, 0.25$ and $0.4m/s^2)$.

5.3 Conclusions

The results presented show how the controller is able to minimize the position error when the vehicle follows the reference trajectory. Nevertheless, when an obstacle is detected an obstacle avoidance manoeuvre must occur. All the tests carried out show that the vehicle begins to separate from the reference trajectory, until it obeys the constraints defined in the controller. The lateral distance defined, $3m$, is not achieved in all the tests due to the necessity of having a detection distance large enough to allow the execution of the manoeuvre. Also, the different tests show that, the higher the longitudinal distance, the sooner the system begins to look forward to fullfill the lateral distance constraints. This is the reason why when the longitudinal distance is $40m$, the vehicle begins to separate from the reference trajectory sooner. This is also visible in the lateral error graphs, where it is slightly bigger in the last two tests.

Besides, the objective of the complete system will determine its final tuning. If the user wants the obstacle avoidance system to disregard the trajectory errors, and ensure the actions taken by the control system are safe enough, the detection distance must be maximized. The best results, in terms of lateral errors, are obtained by doing this.

The lateral tire forces obtained, related to the slip angle, show that there is a significant increase of the latter while the obstacle avoidance manoeuvre is taking place. This occurs due to the need of the vehicle to perform the avoidance manoeuvre. The slip angle β, defined in equations (3.5) and (3.12), increases since the steering wheel angle increases, in order to perform the safety manoeuvre. Nonetheless, even if the increment in β is not constrained, the results obtained are acceptable. Even though big changes in the vehicle's turning occur, this does not imply an unsafe or uncomfortable process for the passenger.

Chapter 6

Conclusions and future lines of action

6.1 Conclusions

The solution proposed by this work is centered in the position and velocity control of an autonomous vehicle, as well as the implementation of an emergency system in order to perform obstacle avoidance manoeuvres. The multivariable MPC controller designed, based on both kinematic and dynamic components of the plant, ensures the safety of the driving in emergency conditions. Results show how the vehicle is able to manouevre in case of detecting static obstacles. When this detection occurs, the vehicle is able to provide a safe lateral distance, in order to properly overtake the obstacle and continue following the predefined trajectory. The time response obtained in the results certifies the benefits of including safety constraints, instead of recalculating of the trajectory when obstacles are detected.

The development of this work hasn't been easy. The selection of the vehicle's model was difficult: the simplicity of a kinematic model implied that the simulated results would not be precise enough. Even if the dynamic model was at the beginning discarded due to its difficult implementation, it was finally the solution for the project. The consideration of the forces provides realism and authenticity to the simulations and the results.

Furthermore, the controller was primarily designed to control the vehicle's position. Given a reference trajectory, it could perfectly follow the latter. Nonetheless, the obstacle avoidance system was hard to merge in the MPC. Constraints had to be redefined, and as a consequence the optimization process changed. Also, the determination of the new constraints was not easy: the computational effort, which was low before the introduction of the obstacle problem, still was needed to be low. However, the constant calculation of distances between obstacles and the vehicle supposed additional estimations for the process. Despite this, after several proposals and hypothesis, a solution which does not imply a high computational effort was found. The use of transformation matrices and position errors for calculating the obstacle constraints was the hardest, yet most successful part of this work.

Results conclude that the linearity of the model and the convexity of the constraints make the MPC problem suitable for obstacle avoidance emergencies. The low computational complexity of each sub-problem (prediction model, control action and avoidance problem) favours the conclusions of the project. As mentioned along this work, the final results depend on the tuning of the different parameters include in the controller. Therefore, the system's tuning ought to be done according to the needs of the user. In connection with the results obtained in the tests performed, better avoidance results are achieved whenever, in case of scenarios where obstacles are present, the obstacle detection distance increases. Also, high velocities enhance this process. Likewise, collision-free trajectories are performed given any possible tuning.

Due to the complexity of obtaining these results in real-time testings, this work is considered successful. It can be called into question that, in certain situations, some constraints and system responses do not satisfy accurately the requirements of the process. Nonetheless, the controller design always rewards the manouevres considered the safest for each situation.

6.2 Future lines of action

Future works related to this project are focused on expanding the context to dynamic obstacles. The study of the controller's response to longitudinal and transversal motions contrary to the vehicle shall be studied. Additionally, investigations on alternative methods to determine the effects of road adherence will be done. The system response, as well as wit the change of dynamic parameters, depends on the characteristics the model takes into account. Therefore, situations where road inclination and friction coefficients, among others, are included would be interesting in order to continue the validation of this work Also, it would be interesting to implement this solution in vehicles where the dynamic and kinematic characteristics vary with respect to this project. Thus, vehicles like trucks where the wheelbase, mass, moment of inertia and adherence coefficients are different to the Sedan characteristics.

In terms of real systems, the controller ought to be validated again with more cases of study. After doing so, the same situations as the ones presented in this work will be tested on the autonomous vehicle available.

Bibliography

[1] K. J. 1, "Smart people, smart money are betting on truck platooning," sep 2015. [Online]. Available: https://www.trucker.com/business/article/21745678/smart-people-smart-money-are-betting-on-truck-platooning

[2] E. Camacho, C. Bordons, and C. Alba, *Model Predictive Control*, ser. Advanced Textbooks in Control and Signal Processing. Springer London, 2004. [Online]. Available: https://books.google.es/books?id=Sc1H3f3E8CQC

[3] E. Biyik and M. Husein, "Damping wide-area oscillations in power systems: A model predictive control design," *Turkish journal of electrical engineering and computer sciences*, vol. 26, pp. 467–478, 01 2018.

[4] Y. Gao, "Model predictive control for autonomous and semiautonomous vehicles," Ph.D. dissertation, UC Berkeley, 2014.

[5] A. Franco and V. Santos, "Short-term path planning with multiple moving obstacle avoidance based on adaptive mpc," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE, 2019, pp. 1–7.

[6] S. Quinlan and O. Khatib, "Elastic bands: connecting path planning and control," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*, 1993, pp. 802–807 vol.2.

[7] W. H. Organization *et al.*, "Global status report on road safety 2018: Summary," World Health Organization, Tech. Rep., 2018.

[8] G. d. E. Dirección General de Tráfico. Ministerio del Interior, "Autorización de pruebas o ensayos de investigación realizados con vehículos de conducción automatizada en vías abiertas al tráfico general," vol. Instrucción 15/V-113, nov 2015.

[9] S. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. Eng, D. Rus, and M. Ang, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 5, no. 1, p. 6, Feb 2017. [Online]. Available: http://dx.doi.org/10.3390/machines5010006

[10] E. Wachter, "Lateral path tracking in limit handling condition using sdre control," *Chalmers University of Technology, Department of Applied Mechanics, Ghotenburg*, 2016.

[11] K. Zhao, C. Wang, G. Xiao, H. Li, J. Ye, and Y. Liu, "Research for nonlinear model predictive controls to laterally control unmanned vehicle trajectory tracking," *Applied Sciences*, vol. 10, p. 6034, 08 2020.

[12] F. Kuhne, W. F. Lages, and J. G. da Silva Jr, "Model predictive control of a mobile robot using linearization," in *Proceedings of mechatronics and robotics*. Citeseer, 2004, pp. 525–530.

[13] T. Raack, “Autonomous vehicle technology: Model predictive control for advanced controls optimization,” Dec 2017. [Online]. Available: https://taylor.raack.info/2017/12/autonomous-vehicle-technology-model-predictive-control-for-advanced-controls-optimization/

[14] P. F., “Model predictive control,” Dec 2017. [Online]. Available: https://fjp.at/control/model/predictive/model-predictive-control/#mpc-algorithm

[15] B. T. Switzer, “Robotic path planning with obstacle avoidance,” Ph.D. dissertation, Rochester Institute of technology, 1993. [Online]. Available: https://scholarworks.rit.edu/theses/789

[16] J. Kong, M. Pfeiffer, G. Schildbach, and F. Borrelli, “Kinematic and dynamic vehicle models for autonomous driving control design,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, 2015, pp. 1094–1099.

[17] L. Tang, F. Yan, B. Zou, K. Wang, and C. Lv, “An improved kinematic model predictive control for high-speed path tracking of autonomous vehicles,” *IEEE Access*, vol. 8, pp. 51 400–51 413, 2020.

[18] “What is truck platooning?” p. 3, 2017. [Online]. Available: https://www.acea.be/uploads/publications/Platooning_roadmap.pdf

[19] C. Bergenhem, E. Hedin, and D. Skarin, “Vehicle-to-vehicle communication for a platooning system,” *Procedia - Social and Behavioral Sciences*, vol. 48, pp. 1222–1233, 12 2012.

[20] “Linear model predictive control.” [Online]. Available: http://www.kostasalexis.com/linear-model-predictive-control.html#

[21] M. A. Abbas, “Non-linear model predictive control for autonomous vehicles,” Ph.D. dissertation, UOIT, 2011.

[22] S. C. Peters, “Optimal planning and control for hazard avoidance of front-wheel steered ground vehicles,” Ph.D. dissertation, Massachusetts Institute of Technology, MIT, 2012.

[23] V. Turri, A. Carvalho, H. E. Tseng, K. H. Johansson, and F. Borrelli, “Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads,” in *16th international IEEE conference on intelligent transportation systems (ITSC 2013)*. IEEE, 2013, pp. 378–383.

[24] F. Lin, K. Wang, Y. Zhao, and S. Wang, “Integrated avoid collision control of autonomous vehicle based on trajectory re-planning and v2v information interaction,” *Sensors*, vol. 20, no. 4, p. 1079, 2020.

[25] C. Hu, L. Zhao, L. Cao, P. Tjan, and N. Wang, “Steering control based on model predictive control for obstacle avoidance of unmanned ground vehicle,” *Measurement and Control*, vol. 53, no. 3-4, pp. 501–518, 2020. [Online]. Available: https://doi.org/10.1177/0020294019878871

[26] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[27] C. O. source simulator for autonomous driving research., “Rendering options, carla simulator docs,” dec 2020. [Online]. Available: https://carla.readthedocs.io/en/latest/adv_rendering_options/

[28] Unknown, “Cobyla,” -, sep -. [Online]. Available: http://zims-en.kiwix.campusafrica.gos.orange.com/wikipedia_en_all_nopic/A/COBYLA

Appendix A

Default Algorithm Parameters

A.1 Prediction Model Parameters

Table A.1: Parameters for the Prediction Model of a Sedan passenger car.

Parameter	**Description**	**Value**	**Units**
I_z	Moment of inertia of the car about the yaw-axis	2800	$kg \cdot m^2$
m	Car mass	1860	kg
lf, lr	Distances of the front and rear tyres to the centre of mass	1.2, 1.5	m
Cf, Cr	Front and rear tyre cornering stiffness	4000	N/rad

In case the velocity is negative (meaning, the car is decelerating), velocity is considered null. Also, the turning angle is normalized: it must be between -π and π.

A.2 Control Action Parameters

Table A.2: Constraints for the control action.

Parameter	**Description**	**Value**	**Units**
$\Theta_{min,max}$	Acceleration range	$[0, 1]$	m/s^2
$\delta_{f-min,max}$	Steering wheel turn range	$[-540, 540]$	deg
$\Delta\Theta$	Acceleration increment	0.05	m/s^2
$\Delta\delta_f$	Steering wheel turn increment	0.1	deg
β	Maximum Slip angle	2.5	deg

A.3 Obstacle Avoidance Parameters

Table A.3: Time parameters for the optimization process.

Parameter	**Description**	**Value**	**Units**
-	Iterations	200	-

Table A.4: Time parameters for the control action.

Parameter	**Description**	**Value**	**Units**
N	Prediction and control horizon	6	*steps*

Table A.5: Obstacle avoidance constraints for the control action.

Parameter	**Description**	**Value**	**Units**
d_{lonmin}	Longitudinal distance veh-obs before avoidance manoeuvre	5	m
d_{lonmax}	Maximum longitudinal distance veh-obs after avoidance manoeuvre	20	m
d_{latmin}	Minimum lateral distance veh-obs	3	m
d_{latmax}	Maximum lateral distance veh-obs	6	m

Appendix B

Article

The work in this thesis has been the base for an article sent to 32nd IEEE Intelligent Vehicles Symposium (IV21), the annual technical forum sponsored by the IEEE Intelligent Transportation Systems Society (ITSS). The paper acceptance is yet to be received.

As this work, the article has been developed with the help of CARLA simulator, by implementing the multivariable MPC controller in order to control the vehicle's position, as well as provide the obstacle avoidance system in case of emergency situations.

Obstacle Avoidance System based on a Model Predictive Controller for Autonomous Vehicles

Carmen Barbero, Iván García[1], Ola Benderious[2] and Rubén Izquierdo

Abstract— **This article proposes an obstacle avoidance system employable in emergencies, which considers the vehicle's physical limitations to minimize any possible damage to the system and the loss of control of the system. The solution proposes a multivariable Model Predictive Controller (MPC) to control the position, orientation and velocity. Some of its constraints are based on the tires' lateral forces, which are evaluated by using a dynamic and kinematic model of the vehicle. This proposal provides reaction times in the range of 50ms, thanks to the lack of trajectory recalculations for avoiding obstacles. The results obtained from the system validation are presented with CARLA simulator for autonomous driving. A static obstacle is detected 10 meters apart, at a velocity of 62 km/h; the avoidance manoeuvre provides an approximate safety distance of 3 meters, without exceeding the vehicle's safety limitations, as it is presented in the results.**

I. Introduction

Over the last 40 years, the field of study of autonomous vehicles has suffered a significant evolution, due to the increment in the power and the reduced cost of the sensors and computer systems [1]. To bring new advances and contributions to this field of study, numerous specialists in the field are developing techniques related to trajectory planning and decision making, to avoid obstacles in an optimal way.

Generally, this problem is solved with a replanning strategy [2]. If an obstacle is encountered within the initial trajectory, the system will hypothesize an alternate path until the optimal solution is found. However, several studies have proposed that the collision avoidance problem can be approached in two ways in terms of control. Indeed, that can be combined by modifying the cost function, or by only adding new constraints (additional penalty functions) to the controller [3].

[4] proposes introducing penalty functions inside the cost function to readjust the vehicle's trajectory, considering the velocity and distance between the vehicle and the obstacle. However, [5] and [6] consider that hazard avoidance capability can be introduced in any navigation system simply through the proper use of the information gathered by sensor, motion and control systems. These solutions propose, among others, the introduction of additional safety constraints for the lateral error and the development of an optimization problem based on the cost minimization under road constraints. Without the recalculation of the vehicle's trajectory, but with the help of constraints inside the control problem, the autonomous vehicle can execute safe trajectories without colliding. [7] proposes the integration of dynamic characteristics as well as road constraints for solving the obstacle avoidance problem, ensuring at all times the safety of every manoeuvre. However, every model is susceptible to failure. The complexity of evaluating most of the vehicle's dynamical parameters (frictions, aerodynamical coefficients) provokes the need to consider many uncertainties to have robust models, as detailed by [8].

Consequently, this work presents a deep study of the mentioned techniques, basing the obstacle avoidance behaviour only on the solutions proposed by the position and velocity controller. An MPC controller based on dynamic features is developed; it considers avoidance constraints (lateral distance to the obstacle, road morphology) and safety constraints for the manoeuvre (tires' lateral forces, slip angles) to drive an optimal control action. However, motion planning and control when hazard avoidance is introduced is challenging for several reasons, primarily trajectory constraints derived from the vehicle physical and stability limits. Also, computation time is a critical parameter which influences the manoeuvre behaviour. Long prediction horizons may improve performance, but with an increased computational demand [7]; also, short prediction horizons can disregard important information for the control analysis. As a result, the verification and validation process is done in simulation environments for safety and economic reasons.

Simulation tools are employed to model the dynamic systems and compare different strategies according to the case of study. Despite the separation of their design strategies, simulation and testing of prediction models and control systems can be tested together. Thus, [9] proves the effectiveness of both designs with the CarSim-Matlab/Simulink co-simulation.

The rest of the paper is organized as follows. Section II describes the dynamic vehicle model. In section III is presented the optimization problem, where it is defined the physical and environmental constraints. Section V, use cases results is presented and discussed with the CARLA simulator. Finally, section VI analyses the main conclusions and future work.

Nomeclature

- $\{XY\}$ inertial coordinate frame
- $\{xy\}$ vehicle coordinate frame

*This work was not supported by any organization

[1]Carmen Barbero and Iván García and Rubén Izquierdo are with the Department of Computer Engineering, University of Alcalá, 28805 - Alcalá de Henares (Madrid), Spain `ivan.garciad@uah.es`

[2]Ola Benderious is with the Department of Electrical Engineering, University of Chalmers, Goteborg, Sweden `ola.benderius@chalmers.se`

- $\dot{x} = v_x$ longitudinal velocity at CG of vehicle
- $\dot{y} = v_y$ lateral velocity at CG of vehicle
- $\ddot{x} = a_x$ longitudinal acceleration of vehicle at CG
- $\ddot{y} = a_y$ inertial acceleration of vehicle at CG
- m total mass of vehicle
- I_z yaw moment of inertia of vehicle
- l_f, l_r longitudinal distance form CG to front (rear) tires
- $l_{fr} = l_f + l_r$ wheelbase
- ψ yaw, heading, angle of the vehicle in global axes
- $\dot{\psi}$ yaw rate and yaw rate of vehicle
- β vehicle slip angle at CG
- δ_f front wheel angle
- $[\alpha_f \; \alpha_r]$ slip angle at front and rear wheel tires
- $[C_{\alpha f} \; C_{\alpha_r}]$ cornering stiffness of front and rear tire
- V vehicle velocity vector

II. Vehicle model

Generally, the behaviour of MPC controllers is dependant on the model of the system to be controlled. In this paper, both dynamic characteristics and kinematic equations model its velocity, position and orientation. The implemented model uses the control actions, steering angle and throttle position, to evaluate the vehicle's state. This state vector of the system is defined by:

$$[x, y, v, \psi, \ddot{y}, \dot{\psi}] \tag{1}$$

Generally, in those autonomous driving applications where high speeds are present, dynamic models provide essential information about the vehicle's state. Also, the forces acting in the system cannot be neglected since their effects affect the motion of the vehicle significantly. Consequently, this paper uses a dynamic modelling of the system by making a derivation based on forces and moments balance with a single-track vehicle's inertial reactions. It assumes that the vehicle is symmetric, roll and pitch movements, and the linear movement in z are neglected. The tire's lateral force is considered to be a linear function, dependent on the slip angle. It is important to note that this model simplification does not consider the coupling between lateral and longitudinal dynamics.

The dynamic model which evaluates the system accelerations is represented by the equations (2a), (3) and (4). This model is based on the behaviour of the vehicle's transmission system behaviour, and the vehicle's behaviour under the effect of lateral forces.

The vehicle's throttle pressure and its state conditions, generate a determined acceleration's velocity. This is directly related to power transmission issues influenced by wheel dimensions, gearbox system, maximum engine torque, tire friction coefficient or aerodynamic characteristics. Differential equation (2a) defines the longitudinal acceleration of the vehicle when the previous parameters are considered.

$$\ddot{x} = G_{Engine} P_t - \mu \, |v_x| - \mu_{aero} \, v_x^2 \tag{2a}$$

$$G_{Engine} = \frac{T_{enginemax} \, r_{eff} \, GR}{J_{engine}} \tag{2b}$$

$$FricCoef = \frac{GR^2 \, r_{eff}^2 \, C_r}{J_{engine}} \tag{2c}$$

G_{Engine} represents the gain that the throttle's position exerts onto the vehicle's acceleration. Also, μ represents the transient state of the acceleration system; this is related with the friction coefficient affecting the system.

For the evaluation of the variables, accelerations and vehicle's yaw rate are initially measured with vehicle's inertial sensors. [10] develops the equations (3) and (4).

$$\ddot{y} = -\frac{2C_{\alpha f} + 2C_{\alpha r}}{m v_x} \dot{y} - (vx - \frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{m * v_x}) \dot{\psi} + \\ + \frac{2C_{\alpha f}}{m} \delta \tag{3}$$

$$\ddot{\psi} = -\frac{2l_f \, C_{\alpha f} - 2l_r \, C_{\alpha r}}{I_z \, v_x} \dot{y} - \frac{2l_f^2 \, C_{\alpha f} + 2l_r^2 \, C_{\alpha r}}{I_z \, v_x} \dot{\psi} + \\ + \frac{2l_f \, C_{\alpha f}}{I_z} \delta \tag{4}$$

The previous equations estimate the translational and rotational dynamics of the vehicle. These are essential to evaluate kinematically the vehicle's position and orientation in the global system of reference. The differential equations (5) define the vehicle's velocity module, by knowing the velocity components v_x and v_y. They allow the position's evaluation $\{X, Y\}$ in the global system of reference, with the Ackerman steering geometry.

$$V = \sqrt{v_x^2 + v_y^2} \tag{5a}$$

$$v_x(t) = a_x(t)dt + v_x(t-1) \tag{5b}$$

$$v_y(t) = a_y(t)dt + v_y(t-1) \tag{5c}$$

$$X(t) = V dt cos(\psi) + X(t-1) \tag{5d}$$

$$Y(t) = V dt sin(\psi) + Y(t-1) \tag{5e}$$

Moreover, the differential equations (6) define the vehicle's turning velocity as well as its orientation, referenced to the coordinate system $\{X, Y\}$.

$$\dot{\psi}(t) = \ddot{\psi} dt + \dot{\psi}(t-1) \tag{6a}$$

$$\psi(t) = \dot{\psi} dt + \psi(t-1) \tag{6b}$$

The dynamic analysis of lateral tire forces defines the slip angle (7) as the difference between the angle of the wheels (directly proportional to the steering wheel angle) and the angle of the vehicle's velocity (decomposed in v_x y v_y, respectively). This parameter is considered essential to determine the sliding behaviour of the vehicle. As a consequence, it must be limited to prevent the vehicle's loss of control. Values greater than $2.5°$ produce, in most tires, a loss of adhesion to the road, causing slipping.

$$\alpha_f = \delta - (\dot{\psi} - \frac{\dot{y}}{v_x}) \tag{7}$$

The parameters used in the model are listed below: those essential for position and orientation, as well as the necessary ones for evaluating velocity.

TABLE I: Vehicle model parameters

Parameter	Symbol	Value
Mass	m $[kg]$	1360
Inertial Moment	I_z $[kgm^2]$	2800
Wheel Base	l_f $[m]$	1.2
Wheel Base	l_r $[m]$	1.5
Cornering Stiffness Forward	$C_{\alpha f}$	2000
Cornering Stiffness Rear	$C_{\alpha r}$	26000
Gear Ratio	GR	2.6
Engine maximum torque	$T_{enginemax}$ $[N]$	400
Wheel radius	r_{eff} $[m]$	0.33
Friction coefficient	μ	0.3
Aero coefficient	μ_{aereo}	0.7

III. Model Predictive Controller Formulation

The MPC controller designed aims to control the vehicle's velocity and orientation by following velocity instructions and reference trajectories, as well as to avoid any obstacle detected in the surroundings of the trajectory. Generally, this problem is solved by replanning the reference trajectory. Despite this, this research proposes a different approach which differs from recalculation. The proposed solution, based on an inherent optimization process with constraints, solves the obstacle avoidance problem by allowing the addition of physical limits to the manoeuvrability of the vehicle. This technique is characterized by providing low time responses, in comparison to processes where an intermediate trajectory recalculation is done. Also, the system manoeuvres taking into consideration the physical limits of the vehicle, in order to avoid any possible loss of the vehicle's control.

The MPC controller contemplates a prediction horizon of 3 seconds, which provokes a dependance between the vehicle's velocity and the predicted distance considered. Supposing velocities are inside the range of $30km/h$ and $60km/h$, the prediction horizon is determined to be between $24m$ and $45m$, respectively, considering the latter enough in case of emergency manoeuvres. Since $dt = 10ms$ in the prediction process, 300 tentative samples are considered in the optimization. In any real time application, high number of tentative samples implies a low speed in the convergence process. Consequently, a technique which involves constraints aggregation is applied; with this, computing time and the optimization solution are improved. Due to the existence of numerous techniques, this work is focused in induction constraint aggregation techniques (*p-norm functionals*).

Equation (8) defines the cost function associated with the MPC controller.

$$J = w_{le}\sum_{i=1}^{N}(y(i) - y_{ref}(i)) + w_{\psi\ e}\sum_{i=1}^{N}(\psi(i) - \psi_{ref}(i)) + \\ + w_{ve}\sum_{i=1}^{N}(v(i) - v_{ref}(i)) + w_{\dot{\rho}}\sum_{i=2}^{N}(\rho(i) - \rho(i-1)) + \\ + w_{\dot{\Theta}_t}\sum_{i=2}^{N}(\Theta_t(i) - \Theta_t(i-1)) + w_{\dot{\alpha_t}}\sum_{i=2}^{N}(\alpha_t(i) - \alpha_t(i-1)) \tag{8}$$

The weights $[w_{le}, w_{\psi\ e}, w_{ve}, w_{\dot{\rho}}, w_{\dot{\Theta}_t}, w_{\dot{\alpha_t}}]$, corresponding to each factor of the cost function, are tuned to follow properly any trajectory reference without obstacles. The lateral distance, orientation and velocity errors committed by the MPC are $30\,cm$, $0.03\,rad$ and $0.2\,m/s$, respectively.

The MPC constraints are described in the following subsections. They enable the vehicle to follow position, orientation and velocity references, as well as to avoid obstacles located in its way, by knowing the lane size and the physical characteristics of the vehicle. The justification for the executed manoeuvres is based on the spatial range which allows the detention of the vehicle (as it has been overpassed). Thus, a pedestrian that is detected too late in a crosswalk, justifies the realization of a safety avoidance manoeuvre. The constraints considered for the optimization process are classified into five types.

A. Lateral distance to the obstacle

The Lateral distance to the obstacle is depicted in figure 1. At any time, the controller must fulfil a minimum lateral distance to the obstacle constraint of $3m$, with an additional tolerance constraint $\pm 0.1m$. The MPC will meet these requirements either to the right or left side of the obstacle. However, there may be yet other obstacles in the environment that can inhibit the actual avoidance solution. For instance, figure 1 depicts how the obstacle will be passed on the left whenever other constraints are violated on the right, such as the road kerb. Equation (9) shows the obstacle lateral distance constraint.

$$\{Tentative_{lateral\ distance}(i)\} \leq 3m\ \forall i \in [1, 300] \tag{9}$$

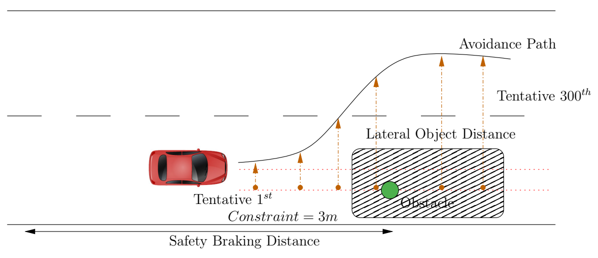


Fig. 1: Obstacle's distance constraint

B. Tire behaviour

The lateral forces suffered by the tire determines the manoeuvre's safety. Therefore, the characteristics of the tires, road conditions, and the vehicle's specific dynamics determine the vehicle's adherence to the road. Wheel slip implies the loss of the vehicle's control. In normal driving conditions, the adherence is appropriate to drive safely. However, whenever an aggressive driving is needed, the adherence is reduced due to the increment of the tires' lateral forces. The slip angle (7), is defined as the difference between the angle determined by the velocity vector and the wheels' steering angle; therefore, a high slip angle results in wheel slip and, consequently, in the vehicle's control loss. Figure 2 shows the tire behaviour in the presence of lateral forces: lateral forces increase lineally as the slip angle rises. Nonetheless, when an absolute slip angle value is achieved, the tires' lateral forces decrease due to the vehicle slip. Thanks to this, the controller can safely manoeuvre. Equation (10a) defines the mathematical expression relating the steering angle and the tire's lateral force. Equation (7) establishes the slip angle behaviour, while (10b) determines the prediction horizon's constraints depending on the tire lateral force. The chosen evaluation method only depends on the tangential acceleration, angular and longitudinal velocity.

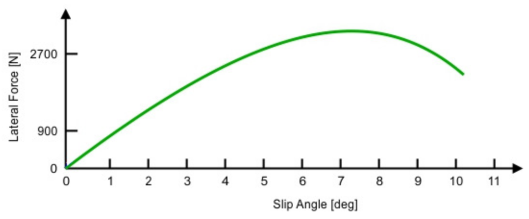


Fig. 2: Lateral force vs slip angle for a normal load of 2000N.

$$F_l = K_v\ \alpha(t) \qquad (10a)$$

$$\{F_l(i)\} \leq F_{l_{peak}}\ \forall i \in [1, 300] \qquad (10b)$$

C. Morphological restrictions of the road

The consideration of morphological road restrictions to manoeuvre is essential in the obstacle avoidance problem. There will be avoidance manoeuvres which may be executable either on the left or the right side of the obstacle; however, physical limits like road kerbs or lane ends will determine the decisive manoeuvre and under which conditions it is performed. In our case, the vehicle's distance to the lane limits is given by the simulator. This functionality, $lane_{limit}(vehicle_{pos})$, is used to evaluate the distance to the limit of the lane, given the vehicle's position. Its output will be used as a constraint, which must be satisfied in the manoeuvre, as depicted in figure 3. The vehicle's centre must be 2 meters distant from the lane limit, described in equation (11).

$$\{Tentative_{lane_{limit}}(i)\} \leq 2m\ \forall i \in [1, 300] \qquad (11)$$

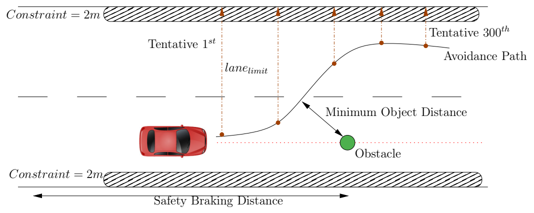


Fig. 3: Road constraints.

D. Physical limitations of the vehicle actuators

The control actions of the vehicle are the steering wheel angle and the throttle's position. Thanks to the change of these variables at each sampling period, it is possible to avoid obstacles in the trajectory surroundings. Nonetheless, the control variables do not have an infinite range of work, but they are mechanically constrained. $\pm 540°$ is the limit of the steering wheel angle, whereas the throttle's position range is $[0, 1]$ (from a null to a maximum pressure in the throttle). Equation (III-D) describes these constraints.

$$-540° \leq \{\sigma(i)\} \leq 540°\ \forall i \in [1, 300]$$
$$0 \leq \{\Theta_t(i)\} \leq 1\ \forall i \in [1, 300]$$

Due to the excessive number of constraints considered in each control period, constraints aggregation techniques are applied. In this way, the controller's computing time is reduced, and the maximum control period is fixed to $100ms$. The aggregation technique used is the Induced Aggregation KS (12) where g_i represents the values obtained by the described variables, and ρ represents the curvature parameter of the constraints aggregation process ($\rho = 100$), to fulfil the properties of the defined aggregation processes described in [11].

$$c_{IE}(\rho) = \frac{\sum_i g_i\ e^{g_i \rho}}{\sum_i e^{g_i \rho}} \qquad (12)$$

IV. Formulation of the Optimization Problem

The optimization algorithm used to minimize the cost function is COBYLA algorithm: a gradient-free optimization algorithm capable of handling nonlinear inequality constraints. The convergence of COBYLA is slow; however, its high stability and the few parameters needed to tune it make it very popular. Equation (13) describes the optimization problem to solve.

$$\begin{aligned}
\underset{\sigma, \Theta}{\text{minimize}} \quad & J(\sigma, \Theta) \\
\text{subject to} \quad & Lateral_{dist}(\sigma, \Theta, i) \leq 3, && \forall i \in [1, 300], \\
& Lane_{limit}(\sigma, \Theta, i) \leq 2, && \forall i \in [1, 300], \\
& F_l(\sigma, \Theta, i) \leq F_{l_{peak}}, && \forall i \in [1, 300], \\
& -540 \leq \sigma(i) \leq 540, && \forall i \in [1, 300], \\
& 0 \leq \Theta_t(i) \leq 1, && \forall i \in [1, 300]
\end{aligned} \qquad (13)$$

V. Results

The results presented in the paper are based on the use of CARLA, an open-source autonomous driving simulator and realistic driving behaviour [12] where two avoidance scenarios are studied. CARLA has been remarkably useful for validating the system, due to the possible damages that can be caused in real systems' testing. This simulator has allowed safe testing and the study of the obstacle avoidance solution under different obstacle positions. The simulator runs on graphic card Nvidia Titan RTX NVIDIA, and the MPC controller runs on an Intel i7 core at 2.9GHz.

The system's behaviour is tested under two situations, marked by the obstacle position on the road. The first case of study positions the obstacle in a straight running lane, the second considers it before entering the curved trajectory. The right lane is free for all the scenarios, and allows the avoidance manoeuvre's execution. Figure 4 depicts the object's position in the simulator environment for the two cases, as well as the environment main characteristics. CARLA integrates inertial sensors to measure the acceleration and angular velocity variables, needed by the controller and avoidance system.

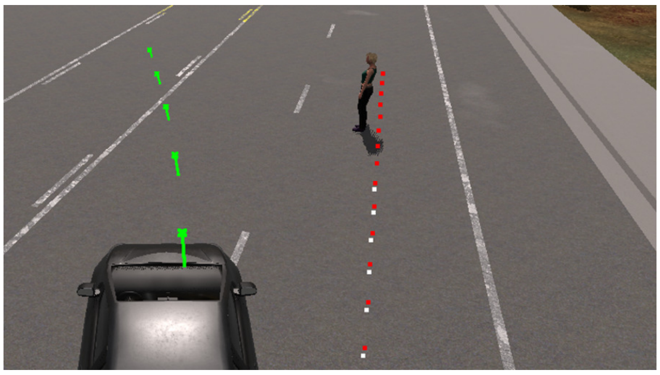


Fig. 4: CARLA case of study.

Three speeds were tested (30, 50 and $65 Km/h$) and three obstacle detection distances (10, 20 and $40 meters$), higher than the emergency braking distance. All the graphs represent the results at the vehicle's centre of mass, considering the static obstacle in the centre of the lane. Figure 5.a depicts the avoidance manoeuvre in a straight trajectory for the different velocities studied, and a detection distance of $40m$. Figure 5.b depicts the vehicle's different avoidance manoeuvres when the vehicle moves at $14m/s$, if the obstacle is placed before the entrance the curved trajectory and the obstacle distance detection are 10, 20 and $40 meters$. All the manoeuvres overpass the obstacle smoothly and safely for both the driver and the obstacle.

Figure 6 compares the tire's lateral forces when the obstacle is detected at the three different distances and a velocity of 17m/s. For every simulation, the maximum force value does not overpass the $4000N$ constraint. As before, the constraint fulfillment guarantees the manoeuvre's safety and provides the vehicle's control. The lateral force increases as the detection distance lowers, due to the aggressiveness of the

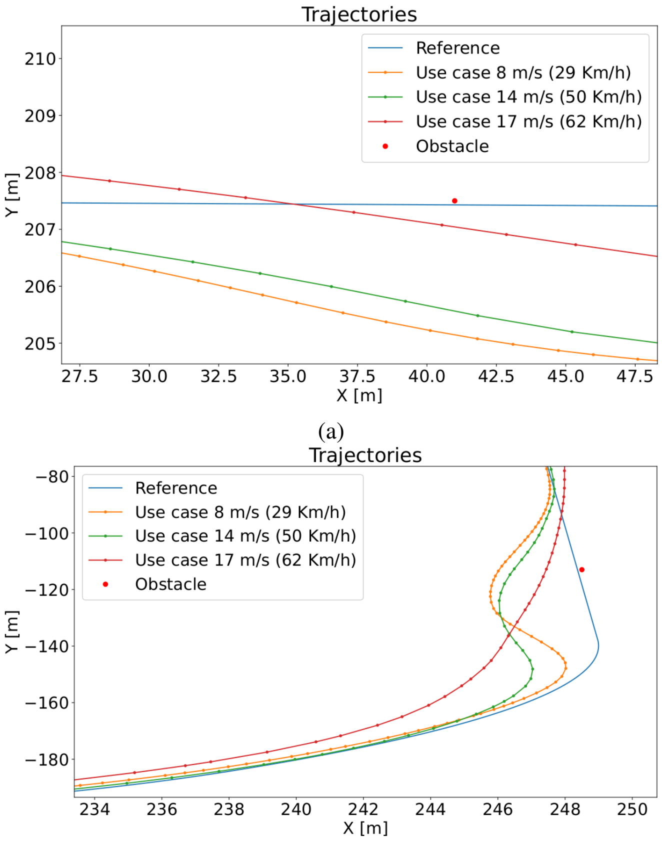


Fig. 5: Obstacle avoidance trajectories under different conditions.

avoidance manoeuvre. When the detection happens at $40m$, the lateral force associated is $1000N$, whereas this force increases to $2000N$ when the detection occurs at $10m$. Lateral force variations after the vehicle's overpass correspond to the manoeuvre needed to return to the reference trajectory. Th return-manoeuvre can be slowed down; however, this work has considered the need to return fastly to the reference. The vehicle's behaviour during the avoidance manoeuvre is shown in the video uploaded.

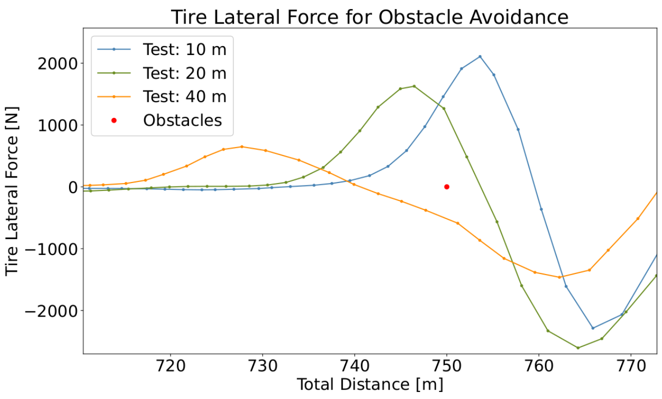


Fig. 6: Tire lateral forces vs. Obstacle detection distance.

On the other hand, figure 7 compares the lateral distance at which the vehicle overpasses the obstacle in the conditions already described. Here, the lateral distance to the obstacle lowers as the detection distance is reduced, due to the need to satisfy the safety constraints, but having less time. When the detection distance equals $40m$, the lateral distance corresponds to $3.3m$, whereas the latter is reduced to $2m$ when the detection occurs at $10m$.

It can be called into question the fact that minimum constraints are not fulfilled in the last case. Nonetheless, the controller design always rewards those manoeuvres which are safe, to avoid the vehicle's loss of control. The video shows that, since it is a multivariable control system, velocity lowers when the avoidance manoeuvre starts. This facilitates the system's constraints fulfillment. Nonetheless, all the cases of study proposed result in similar plots depicting the return to the reference trajectory; this derives from applying the same constraints in the optimization process. Being able to perform a safe manoeuvre manually at $62Km/h$ in order to avoid an obstacle located $10m$ apart, is a complex and precise task even for experienced drivers. As a consequence, the results obtained are considered a significant achievement.

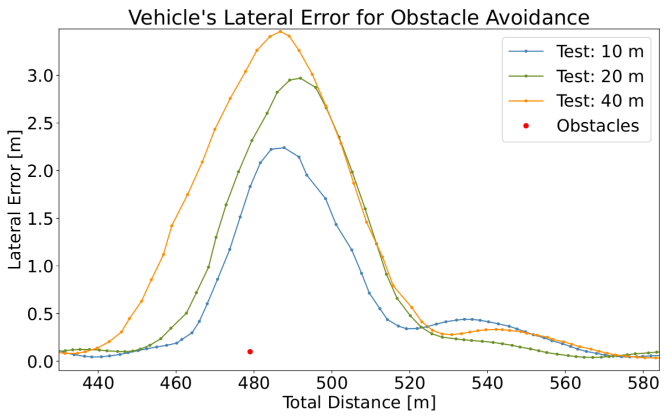


Fig. 7: Lateral distance to the obstacle vs. Obstacle detection distance.

VI. Conclusions and Future Works

A validation of a system based on a multivariable MPC controller is done, to safely avoid obstacles present in autonomous driving scenarios. Emergency conditions involve high velocities as well as proximity to the obstacle. Therefore, the solution proposed is an optimal solution to avoid road obstacles in emergency conditions by using an MPC controller that integrates tire lateral forces, to achieve limits where the manoeuvre's slipping conditions drive the vehicle to work in its safety limits. The results show how a static obstacle detected at $10meters$ at $62km/h$; the avoidance manoeuvre provides an approximate safety distance of $2.5meters$, without exceeding the vehicle's safety limitations.

These obstacle avoidance techniques, applicable due to the controller constraints, reduce the system's reaction time to $50ms$, coinciding with the control cycle; this improves the time response of those systems that have to, simultaneously, recalculate the trajectory to perform a collision-free course.

In the future, this work aims to be applied to dynamic obstacles following movements that are longitudinal and transversal to the vehicle's motion, and an analysis of the controller's avoidance behaviour. Additionally, the effect of road slipping in the system ought to be studied; for this, cases of study involving different road adhesion coefficients are needed. Future works will focus on two lines of action. Firstly, after the simulation testing, the controller will be tested in a real system, in a safe environment. The same simulated cases of study will be compared with the real results. On the other hand, a hardware in the loop environment will be developed to help validating the real system.

Appendix

The MPC controller has been totally implemented in Python, source code is available at the following Git repository: https://github.com/ivangarciad/mpc_avoidance.git.

Acknowledgment

This work was supported in part by grant CM/JIN/2019-009 (CAM), project to promote the creation and consolidation of young researches, by Research Mobility Grant that Iván García obtained in 2019 by the University of Alcalá and in part by grant 723021 (BRAVE, H2020).

References

[1] C.-Y. Chan, "Advancements, prospects, and impacts of automated driving systems," *International Journal of Transportation Science and Technology*, vol. 6, no. 3, pp. 208–216, 2017, safer Road Infrastructure and Operation Management. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2046043017300035
[2] B. T. Switzer, "Robotic path planning with obstacle avoidance," Ph.D. dissertation, Rochester Institute of technology, 1993. [Online]. Available: https://scholarworks.rit.edu/theses/789
[3] G. Ferrer Usieto, "Trajectory generation for autonomous highway driving using model predictive control," Master's thesis, Universitat Politècnica de Catalunya, 2017.
[4] Y. Gao, "Model predictive control for autonomous and semiautonomous vehicles," Ph.D. dissertation, UC Berkeley, 2014.
[5] S. C. Peters, "Optimal planning and control for hazard avoidance of front-wheel steered ground vehicles," Ph.D. dissertation, Massachusetts Institute of Technology, MIT, 2012.
[6] V. Turri, A. Carvalho, H. E. Tseng, K. H. Johansson, and F. Borrelli, "Linear model predictive control for lane keeping and obstacle avoidance on low curvature roads," in *16th international IEEE conference on intelligent transportation systems (ITSC 2013)*. IEEE, 2013, pp. 378–383.
[7] J. Wurts, J. L. Stein, and T. Ersal, "Collision imminent steering using nonlinear model predictive control," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 4772–4777.
[8] J. Wurts, J. Dallas, J. L. Stein, and T. Ersal, "Adaptive nonlinear model predictive control for collision imminent steering with uncertain coefficient of friction," in *2020 American Control Conference (ACC)*, 2020, pp. 4856–4861.
[9] H. Wang, B. Liu, X. Ping, and Q. An, "Path tracking control for autonomous vehicles based on an improved mpc," *IEEE Access*, vol. 7, pp. 161 064–161 073, 2019.
[10] R. Rajamani, *Vehicle Dynamics and Control*, 01 2006.
[11] G. J. Kennedy and J. E. Hicken, "Improved constraint-aggregation methods," *Computer Methods in Applied Mechanics and Engineering*, vol. 289, pp. 332–354, 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0045782515000663
[12] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.