Model Predictive Control System Design of a passenger car for Valet Parking Scenario

by

Oussama Erraji

A Thesis
Submitted to the Faculty of Graduate Studies
through the Department of Mechanical, Automotive, and Materials Engineering
in Partial Fulfillment of the Requirements for
the Degree of Master of Applied Science
at the University of Windsor

Windsor, Ontario, Canada
2018

Model Predictive Control System Design of a passenger car for a Valet Parking Scenario

by

Oussama Erraji

APPROVED BY:

_____

J. Defoe
Department of Mechanical, Automotive, and Materials Engineering

_____

J. Ahamed
Department of Mechanical, Automotive, and Materials Engineering

_____

N. Kar
Department of Electrical and Computer Engineering

_____

B. Minaker, Advisor
Department of Mechanical, Automotive, and Materials Engineering

September 4, 2018

Declaration of Originality

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices.

Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

Abstract

A recent expansion of passenger cars' automated functions has led to increasingly challenging design problems for the engineers. Among this the development of Automated Valet Parking is the latest addition. The system represents the next evolution of automated system giving the vehicle greater autonomy: the efforts of most automotive OEMs go towards achieving market deployment of such automated function. To this end the focus of each OEM is on taking part to this competitive endeavor and succeed by developing a proprietary solution with the support of hardware and software suppliers. Within this framework the present work aims at developing an effective control strategy for the considered scenarios.
In order to reach this goal a Model Predictive Control approach is employed taking advantage of previous works within the automotive OEM in the automated driving field. The control algorithm is developed in a Simulink® simulation according to the requirements of the application and tested; results show the control strategy successfully drives the vehicle on the predefined path.

بسم الله الرحمن الرحيم

To my Mother, for her love in life and beyond.

Acknowledgements

The present work represents the result of a long journey of technical educational, personal growth and life experience that sparkled five year ago at the Politecnico di Torino. To be the recipient of this life time opportunity to study and develop as a engineer and more importantly as a person during this year long stay in Canada felt to be the greatest of opportunities and indeed a privilege.

Therefore I would like to express my deepest gratitude to everyone that provided us with this invaluable opportunity, from the Politecnico di Torino and University of Windsor to FCA organizations both in Italy and North America. I should also thank for their relentless and positive support my industrial advisor from CRF Dr. Pandeli Borodani, my academic advisor at University of Windsor Dr. Bruce Minaker and our industrial supervisor Marie Mills from ARDC.

Special thanks need to go to the kind and supportive Prof. Giovanni Belingardi whose personal effort was key to the success of the whole project.
In addition to this I would very much like to extent my friendly gratitude to all the people that I shared this year long life changing experience with, from the “Josephine Gang” boys and girls to the “Bridge guys” for making every day fulfilling and unforgettable.

At last I am expressing by deepest gratitude to my Family for their loving and continuous support and my Father in particular, to whom I am eternally grateful for all his sacrifices that made my education and achievements possible.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ADAS	Advanced Driver Assistance System
ADS	Automated Driving System
AVP	Automated Valet Parking
CG	Center of Gravity
CRF	Centro Ricerche Fiat S.C.p.A.
DDT	Dynamic Driving Task
DGPS	Differential Global Positioning System
EPS	Electric Power Steering
FCA	Fiat Chrysler Automobiles
FOV	Field Of View
GPS	Global Positioning System
LED	Light Emitting Diode
LiDAR	Light Detection And Ranging
LQR	Linear Quadratic Regulator
LRC	Lane Recognition Camera
LTI	Linear Time Invariant
MIMO	Multi Input Multi Output
MPC	Model Predictive Control
ODD	Operative Design Domain
OHY	Object Hypothesis
PID	Proportional Integral Derivative
RADAR	Radio Detection And Ranging
RRT	Rapidly exploring Random Tree
SAE	Society of Automotive Engineers
SLAM	Simultaneous Localization And Mapping
SUV	Sport Utility Vehicle
V2I	Vehicle to Infrastructure

List of Symbols

T_r	Rise Time
p	Prediction Horizon
T_s	Sample Time
w	Cost Function Weight
e	Prediction/Tracking Error
Δu	Predicted Input Variation in each Time Step
J	Cost Function
k	Time Step
q	Vehicle's CG offset to Centerline
V_x	Vehicle Longitudinal Speed
V_y	Vehicle Lateral Speed
a_y	Vehicle Lateral Acceleration
D_{vol}	EPS Steering Angle Output
$\dot{\psi}$	Yaw Rate
I_z	Vehicle Inertia around Z axis
$C_{(\alpha f;\alpha r)}$	Tire Cornering Stiffness (front, rear)
l_f	Front Axle – CG Distance
l_r	Rear Axle – CG Distance
m	Vehicle Mass / Centerline Slope
K_l	Road Curvature
L	Look Ahead Distance
R	Radius of Curvature
t	Time
y_{fb}	Look Ahead Prediction of q
T_{dr}	EPS Input Torque
δ	Front wheels steering angle

1 Introduction

The demand for ever increasing automation in the vehicle has been a strong catalyst for automotive innovation in recent years and the trend is constantly on the rise. As sensor technology continues to evolve providing the market with highly capable detection systems, at profitable prices, and software algorithms are taking advantage of increasingly efficient hardware architectures, as the one in the figure below, the automakers find themselves competing to develop more and more advanced ADAS systems while striving to be the first to the market. On the other hand, customer expectations generate more pressure on system engineers to design sophisticated functionalities for drivers and passengers' comfort and safety. Within this market scenario it is clear to understand the potential of a system that is capable of fully automating the driving task: automated valet parking or AVP.

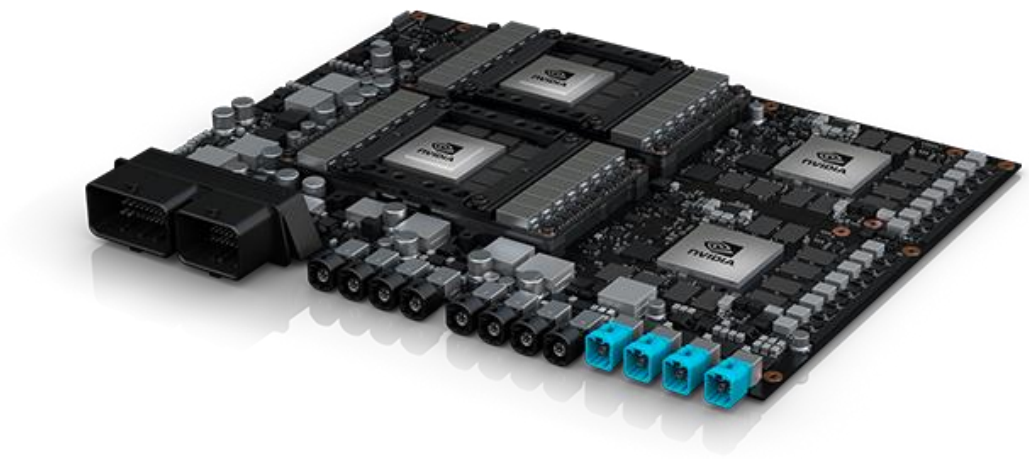

***Figure 1.1**: NVIDIA DRIVE Pegasus offers a deep learning and neural networks architecture for self driving applications [1].*

The system is engaged by either an on-board driver or from outside the vehicle, from a dedicated area such as a drop off area, from where the car departs and engages in the parking task autonomously. During all phases of the parking mission the vehicle should provide informative feedback to the driver, especially in case the task was not achieved, and the car was unable to park itself as a consequence. This may happen due to different reasons. primarily impeding obstruction to the vehicle motion or unavailable parking space in the surrounding environment.

1.1 AVP System outline

The objective of an AVP system is that of accomplishing the task of automatically parking and driving back the vehicle in a given environment while maneuvering in a safe and predictable way. This identifies the AVP system as an on-board Automated Driving System (ADS) that allows the vehicle to perform all driving tasks and monitor the driving environment, essentially doing all the “driving” in given circumstances. Although the human does not need to pay attention and may not even be present at all during the automated maneuvering of the vehicle, the system in case of fallback would still try to involve the driver (e.g. by means of prompt messages on a smartphone) and if he or she is not available the system performs the so-called minimal risk maneuver. This involves slowing down and moving to the right possibly past any crossroads.

AVP systems can be implemented according to two different approaches. The first approach is that of an “intelligent vehicle” in which the whole system is deployed entirely on the vehicle: sensors, processing units and control algorithms are present only on the car. A second approach is known as “intelligent infrastructure” where the system perception and planning function are centrally managed by the infrastructure, which communicates with the vehicle thru a V2I protocol. The control is still present on the vehicle, which also receives information about any obstacle or traffic.

The two approaches have different requirements concerning the system performance and robustness. While in vehicle applications the focus is on the on-board systems complexity and more importantly on the built-in robustness; on the other hand, infrastructure implementations need large investments, and in some solutions also accurate and up-to-date HD maps need to be generated. As it is presented later in this work, both of these schemes have been explored in recent years, yielding very successful solutions, even if limited to the horizon of application.

In order to achieve its goal, the AVP system must be comprising of different functionalities, which dynamically interact with each other to perform the requested task. These can be broken down into three main sub-systems. The first is a perception module, which allows the vehicle to sense the environment around it and collect as much usable information as possible. All this information needs to be processed and made available to the next sub-system in the pipeline while minimizing the delays. The perception module may also include a data fusion stage in which information from different sources is processed together to infer new information or compensate for the lack of it. This will be further analyzed in detail in the next chapter.

***Figure 1.2**: Example of currently available detection technologies [2]*

The second module in the system is the path generation algorithm: arguably, it represents the core of the system logic and the main contributor to its performance. It is clear how the ability of the vehicle to plan a drivable, obstacle free path will determine whether the automated parking task will be accomplished.

Usually the path planner can be included within a multi level motion planner, which schedules different trajectory algorithms depending on the stage of the parking process. An example can be splitting the motion between an initial “scouting” driving mode, during which the car drives around a parking area while looking for free parking spot. When one of these is detected a secondary parking maneuver planner is activated which plans the final path to the goal destination.

The third module includes the control strategy which task is to autonomously drive the vehicle along the planned path while ensuring safety and performance requirements, in terms of accuracy and robustness of the control action, are fully met.

An essential task that the vehicle needs to perform at all stages is localization. Traditionally this has been accomplished by the use of GPS or the more accurate DGPS [3], which provides location accuracies of around 10 cm; yet in autonomous driving application, the need for higher accuracies and the capability of locating the vehicle in GPS denied environments, such as underground or multilevel parking, growing the need for perception-based localization techniques. Among these, odometry is widely used, which consists in obtaining information from onboard vehicle sensors (e.g. vehicle speed, steering angle, wheels angular speed etc.) to reconstruct the vehicle position over time from the initial position. This method can suffer from low accuracies, which are compensated with instrument calibration and refined estimation algorithms. Arguably, the most successful technique is simultaneous localization and mapping or SLAM. The algorithms developed within this second solution allow the vehicle to sense the environment around with the aim of creating a virtual map and to locate itself in it.

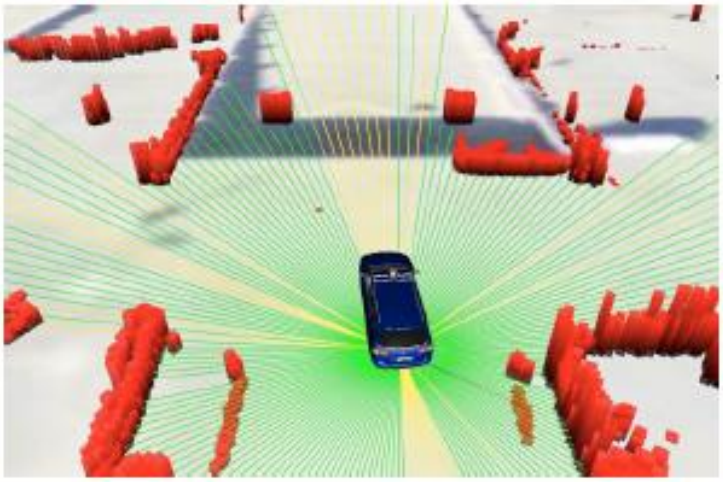

***Figure 1.3**: 2D SLAM generated map in a parking setting [4]*

1.2 Thesis objectives and related project

The ultimate goal of this work is to develop a complete automatic vehicle control strategy for AVP application: the main phases of which are discussed in the following.

The first aspect to consider is the definition of the system architecture, which integrates the vehicle models, the perception module and the control strategy.

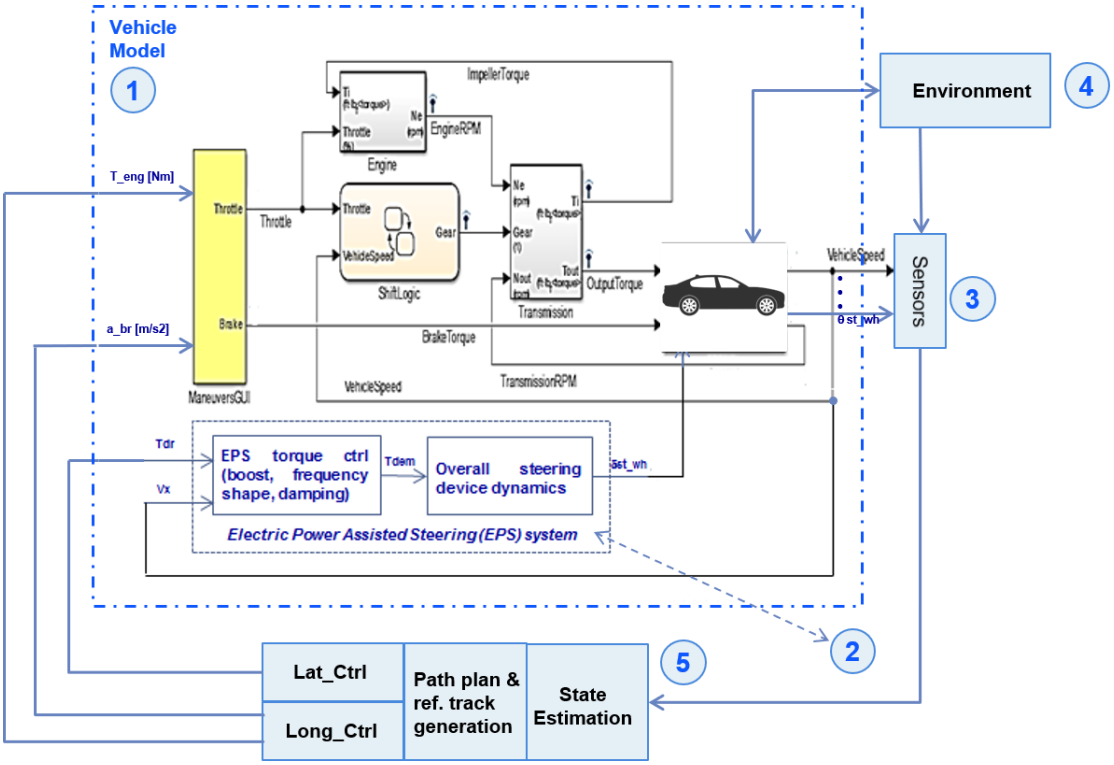

***Figure 1.4**: Functional System Architecture*

The above will be a reference during all stages of the system development. Initially a parking scenario needs to be defined in a software environment from which all following steps of AVP development will follow.

The scenario consists of an area such as a large open space parking lot, a road side parking or a home garage. It is important to properly define the scenario, or a multitude of scenarios, as to best represent real parking scenarios where the system would be engaged: the geometries of the road, traffic rules and parking lot layouts all need to be carefully defined.

The following project phase involves the definition of a sensor set to be used in the simulations and later on, during the validation, on the vehicle itself. The idea is to

compare the performance of a “low-cost” minimal set of sensors to that of a full-set of sensors. The latter would include surround view and front cameras, ultrasonic sensors, front radar and LIDAR sensors: the aim is to study the robustness and capability of a low-cost but ready to market set of sensors as compared to a defined “best performance” benchmark.

After the sensor set is defined, a path planning algorithm needs to be defined within the software environment. Once a trajectory is available the aim will be to design a control strategy for lateral and longitudinal dynamics at low speed driving, this being the main objective of the present work.

At a later stage the complete functional system will be validated in a real AVP scenario using a series production car.

1.3 Methodology

The design of the control system is developed starting from a reference control layout. A lateral and longitudinal dynamics models are provided from CRF (Centro Ricerche Fiat) S.C.p.A modeling the full dynamics passenger car as MATLAB® S-Functions.

These models were identified from test runs data and are provided as ready to use black-box models: a compatible version of MATLAB (2015 32-bit) was necessary as the functions were compiled in such software release.

A simplified overview of the control strategy is presented in the following.

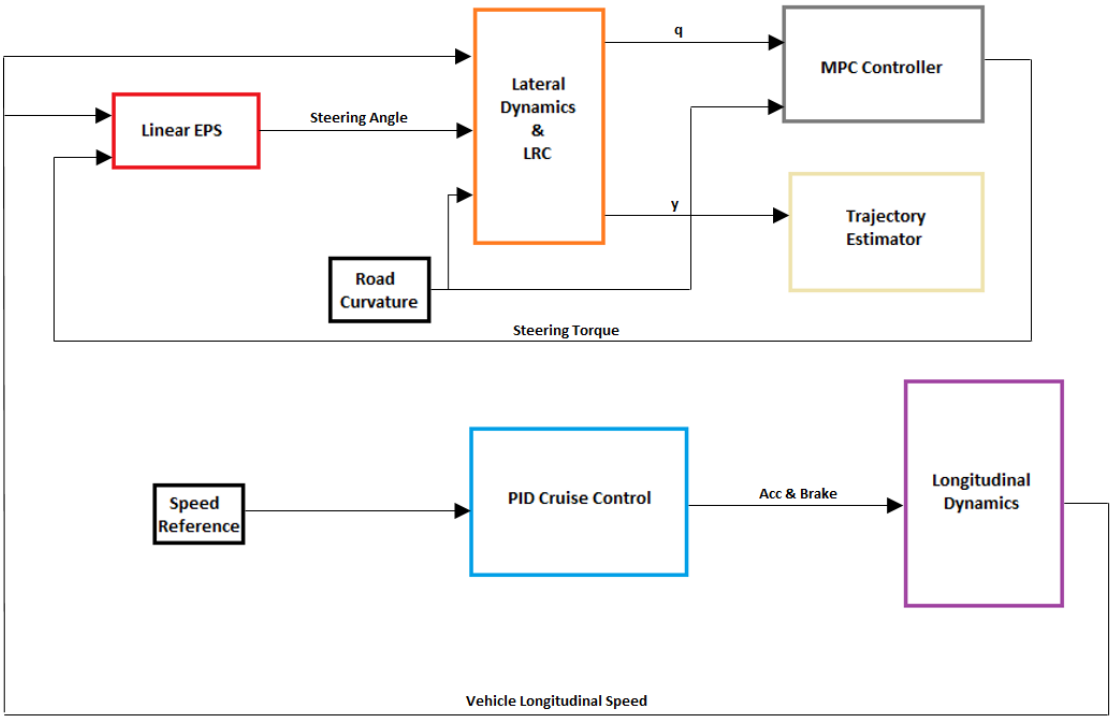

***Figure 1.5**: Simplified control strategy model*

The main inputs to the system are the speed reference, which can be arbitrarily chosen or derived from the path geometry. A PID controller allows following the chosen speed by minimizing the tracking error.

A linear EPS model, obtained from previous works within the OEM's with the use of system identification, receives the control torque input from the MPC controller and generates the steering wheel angle. This, together with the longitudinal speed is fed to the lateral dynamics block, which then yields the key lateral variables of the vehicle: yaw rate, lateral speed and lateral acceleration.

The system also uses a Lane Recognition Camera like system to estimate the vehicle center position relative to the trajectory: this represents the chosen input to the controller along with the road curvature as it is better explained later on.

Finally, the actual trajectory driven by the vehicle is reconstructed by the trajectory estimator starting from the vehicle states, the planned trajectory and the vehicle center of gravity offset to the latter (q).

2 Literature review – State of the Art

Automatic Valet Parking system represents the very next technological step towards fully autonomous vehicles by completely automating the parking task, which still appears to be one of the most demanding driving maneuvers. This system can relieve drivers not only from the task of parking the vehicle yet also from the time-consuming task of finding an empty parking space. Moreover, a great potential benefit of wide use of AVP systems is the possibility of high-density parking (HDP [37]), where autonomously parking vehicles can be rearranged in layouts that optimize parking space utilization while taking advantage of smaller parking slots as vehicle accessibility is not needed. As such, AVP systems are expected to appear in the car market very soon owing to their potentially great customer value.

The vehicle, once the driver engages the system, would be capable of low speed cruising while sensing the surrounding environment for available parking slots and nearby obstacles. Once a free parking slot is detected, the autonomous vehicle will plan a parking path according to the available space and kinematic limitations to drive itself to the selected parking space.

With this definition, AVP qualifies as a level or category 4 driving automation standard according to J3016 SAE [5], meaning that the vehicle is in charge of the whole task once the system is engaged by the driver with the latter not expected to intervene in any situation.

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	**No Automation**	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	**Driver Assistance**	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	**Partial Automation**	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	**Conditional Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	**High Automation**	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	**Full Automation**	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

***Table 2.1**: SAE International's standard J3016-High driving automation levels definition [5]*

***Table 2.2**: SAE International's standard J3016-High driving automation definition [5]*
ODD: Operative Design Domain DDT: Dynamic Driving Task

Level 4 - High Driving Automation

Driver/dispatcher (while the *ADS* is not engaged):
• Verifies operational readiness of the *ADS-equipped vehicle*
• Determines whether to engage the *ADS*
• Becomes a *passenger* when the *ADS* is engaged only if physically present in the *vehicle*

Passenger/dispatcher (while the *ADS* is engaged)*:*
• Need not perform the *DDT* or *DDT fallback*
• Need not determine whether and how to achieve a *minimal risk condition*
• May perform the *DDT fallback* following a *request to intervene*
• May request that the *ADS* disengage and may achieve a *minimal risk condition* after it is disengaged
• May become the *driver* after a requested disengagement

ADS (while not engaged):
• Permits engagement only within its *ODD*

ADS (while engaged):
• Performs the entire *DDT*
• May issue a timely *request to intervene*
• Performs *DDT fallback* and transitions automatically to a *minimal risk condition* when*:*
• A *DDT performance-relevant system failure* occurs or
• A *user* does not respond to a *request*
• A *user* requests that it achieve a *minimal risk condition*
• Disengages, if appropriate, only after:
• It achieves a *minimal risk condition* or
• A *driver* is performing the *DDT*
• May delay *user*-requested disengagement

This also includes scenarios in which during the parking maneuver or even low-speed cruising, a pedestrian or vehicle is detected as an obstacle on the vehicle's path: the system would be able to re-plan a new route avoiding the obstacle or to abort the parking maneuver altogether if no drivable path was found.

For convenience, the literature review will be subdivided in three separate topics that reflect the AVP system architecture layout: perception, path planning and vehicle longitudinal and lateral control.

A comprehensive review of the state of the art technology in AVP applications is present in [16] with some of the publications mentioned further analyzed in the next sections of this chapter.

2.1 Perception and Data Fusion

The first consideration to make when developing an AVP system is the definition of the sensor set: this will have great effect on both the performance and cost effectiveness of the system as an available option for final series production vehicles. As such much of the state of the art research in the AVP system focuses in refining the implementation of sensor sets which include: cameras, ultrasonic, LiDAR and less often RADAR. The chosen set must provide high reliability and high detection accuracy of obstacles and road topology, for any given environmental condition or noise (i.e. glare, fog, dust, dirt, night etc.).

Each typology of sensor has it own advantages and limitation as listed in the following table:

***Table 2.3**: Features that can be traced by different sensors and their drawback [6]*

Sensor Type	Tracking Features	Drawback
LIDAR	Edge target, Range	Higher cost and affected by weather
Video Camera	Vision target (edge, patches, gradients)	Limited depth perception and distance; affected by weather
Infrared Camera	Vision target (edge, patches, gradients)	Non-thermal signatures cannot be detected
Radar	Point target, Range	Limited information about object, higher cost

Camera sensors are usually used to infer appearance of objects either by feature-based techniques or by processing the images using stereo motion algorithms to extrapolate location and possible relative speed; LiDAR can improve the accuracy of shape and dimension sensing while RADAR is effective in estimating relative speed to a detected object.

Compared to other sensors, LiDARs are more expensive, yet they provide more accurate depth estimations; cameras, on the other hand, provide the highest resolution and range among all sensors [37].

Owing to their limited range (<10m) ultrasonic sensors are employed in many parking assist applications for close object detection: they tend to be cheap and weather resistant (snow, fog, rain, glare or darkness).

The sensor set performance is enhanced by implementing data fusion techniques: this allows obtaining more complete and accurate information. Three main typologies of sensor fusion are:

-competitive, in which each sensor provides independent measurements of the same element to increase robustness of the measurement

-cooperative, where information from different sensors is used to infer new information

-complementary, within which the same kind of information is combined to complete existing information.

Sensor information fusion can be made at either feature level, which means using raw data from the sensors or at object level from the obtained object hypothesis:

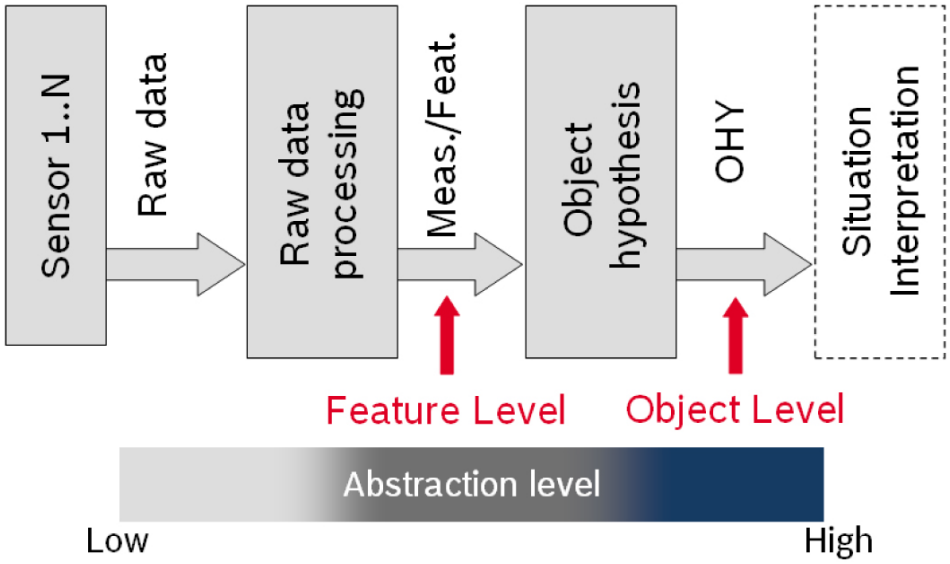

***Figure 2.1**: levels for sensor data fusion. OHY: Object hypothesis [7]*

In [33] the fusion between a set of 12 ultrasonic sensors and a front camera can be found: researchers were seeking to develop a system architecture with the aim of using market ready sensors only in order to reduce cost.

The fusion algorithm allowed enhanced robustness and reliability of the system in real conditions: changing weather, changing lighting conditions and dynamic obstacles avoidance.

Valet parking implementations also include the use of infrastructure or more specifically map based solutions [9] [13] [20] [22] [35] [36] in which the parking area is mapped in separate runs and accurate maps are provided to the vehicle through a V2I communication protocol, along with free parking spot information [26]. This approach is particularly effective in indoor parking areas with limited GPS information, but relies on the availability of updated and accurate maps.

Within these kind of implementations [10] [25] the use of external sensors is also being explored: an infrastructure-based camera system will provide position information to the vehicle. The parking spot will be automatically assigned to the vehicle upon entrance to the parking area where the driver leaves the vehicle in a designated hand-over area. The system also informs the vehicle about detected pedestrian and moving vehicles for collision avoidance.

Vehicle only implementation of AVP systems have been also widely researched with some of these using camera-based applications for parking space detection, as [11] [12] [13] [31] [40], while in others front facing LiDAR [8] [20] [22] [23] [24] [30] [36] sensors were employed, along with ultrasonic sensors for close obstacle detection. In many of the aforementioned applications, as in [20], kinematic-model based odometry was included in the localization task to improve accuracy.

In [12] stereo motion was utilized to provide both a Parking Slot Detection along with Augmented Parking functionality that uses image-based rendering to compute a virtual bird’s eye view; this study compared camera-based methods with feature based approaches and ultrasound-based solution and found that the latter had a limited performance in detecting cross parking slots. The study also showed that top mounted lateral cameras provided greater error of measurements.

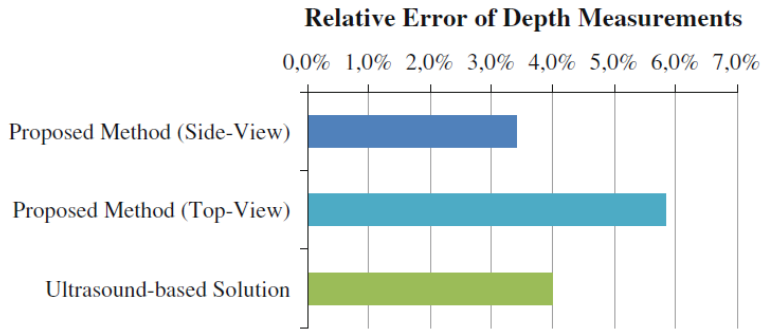

***Figure 2.2**: Parking slot depth measurement error results in [12]*

A Similar vision-based approach was adopted in [11] [31], where through dense stereo motion applied on inverse perspective images, the generation of height maps was conducted; parking space detection is done by using a line extractor on low-obstacles edges, after which space recognition is validated by a probabilistic approach.

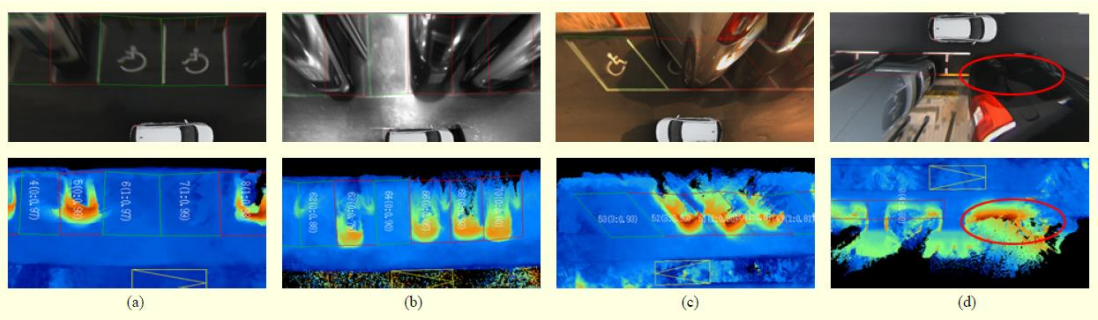

***Figure 2.3**: Input images (above) and recognition results overlapped by Height Map for 4 different parking scenarios and environmental conditions [11]*

This method is robust against adverse environmental conditions but was limited by lack of road sign recognition to detect disabled parking space.

Vision-based solutions in AVP systems are the best equipped to integrate road sign and marking recognition to avoid the ego-vehicle parking itself in tow-away zones, driveway entrances or reserved parking spaces.

Funded by the European Commission the V-Charge project [14] aimed at developing an AVP system for an electric vehicle using series production only sensors: the idea was to include this function on series electric vehicles to guide the vehicles into charge ready parking slots. The vehicle used a set of 12 ultrasonic sensors, a front stereo camera with a 3D measurement range of 50 meters and 4 fish-eye cameras with 185° FOV and 1.3Mpx resolution. Ultrasonic sensors had a 60° horizontal angle and a 30° vertical angle with a maximum range of 4.5m.

With this sensor set, using in particular images generated from the fish-eye cameras, the team of researchers was able to develop algorithms capable of detection and tracking of vehicles, passengers and parking slots reaching state of the art accuracy and reliability [38].

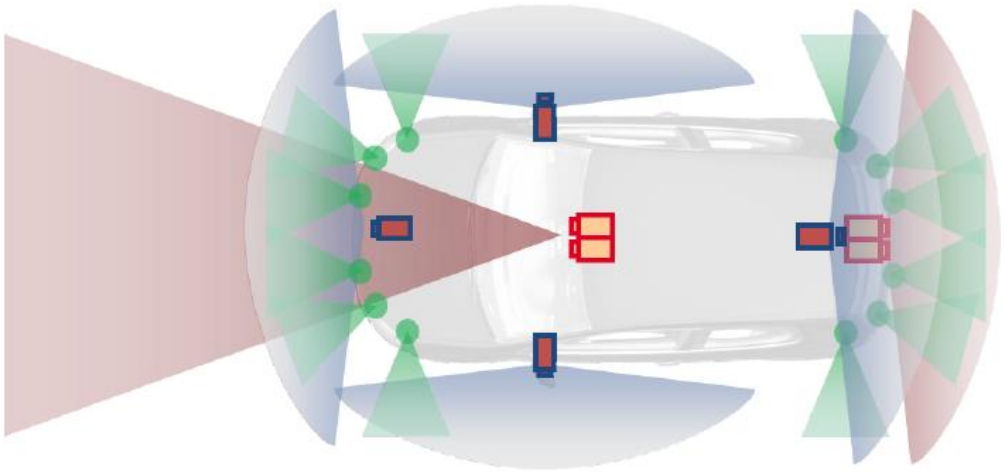

***Figure 2.4**: Sensor set used in V-Charge project with highlighted FOV: green for ultrasonic, blue for fish-eye cameras and red for stereo camera. [14] and [38]*

Within the V-Charge project application three different types of maps were generated: sparse maps containing sparse 3D points used for localization, dense maps used for height definition of surroundings needed for path planning task and finally road graphs containing information regarding features like lanes, parking spots etc.

These are detected by obtaining an overhead image of the environment from the dense map and then running a template matcher to detect predefined parking spots.

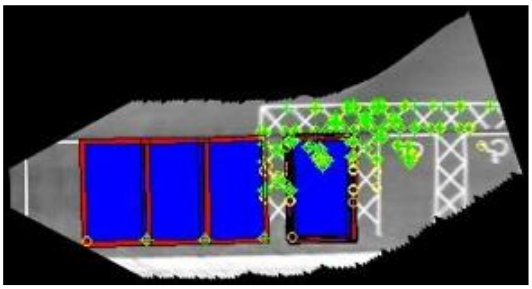

***Figure 2.5**: Parking spot detection result using the template matching technique [14]*

Researchers in [40] used a camera only sensing vehicle guided by external LEDs waypoints to perform valet parking in a indoor environment: the front facing camera provided localization while the rear view one provided parking spot detection.

2.2 Path Planning

The second step in the AVP process is to define a drivable path from the initial position of the vehicle, when the system is engaged by the driver, to the detected free parking spot. Algorithms have been developed already for Park Assist systems which deal with the final parking maneuver, yet the AVP scenario includes also a low speed cruising mode, during which the vehicle is sensing the environment to detect available parking spots while avoiding possible obstacles on its path.

Different constraints apply to the AVP path planning algorithm, which needs to consider: non-holonomic vehicle kinematics, static and dynamic obstacles avoidance, final parking maneuver definition and traffic flow direction.

In order to deal with these requirements, algorithms based on A* [18] [23] [36] and Dijsktra's [17] [21] [35] can be defined including cost functions which aim to optimize the path also from a control point of view. An application of rapidly-exploring random tree (RRT) was found in [8].

The geometry of the path is assembled using straight lines and particular curves such as Dubin's Curves [16] [19], Bézier curves [25] [26] and Reeds-Shepp curves [24] [30], the latter obtained from Dubin's curves but considering also backward maneuvers. B-spline paths have also been tested even if more optimization would be needed as the drivability of this curves may not be guaranteed [20]. A Dijsktra based algorithm in [17] is capable of path planning while considering several safety constraints.

Owing to their simple geometry, circular arches connected by common tangent, Dubin's curves are employed to generate obstacle avoiding paths with a good real-time performance. In [16] a Tentacle Algorithm generates a set of backup paths around the obstacle; a cost function is added considering the distance to obstacle, offset from the ending point to the origin point and a motion cost related to the difficulty of changing the vehicle trajectory.

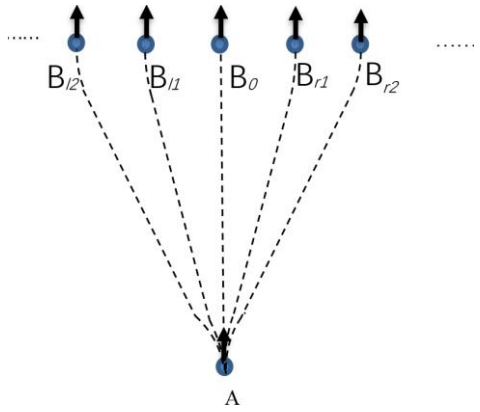

***Figure 2.6**: Backup target positions and paths in Tentacle Algorithm [16]*

If no feasible path is being found the vehicle would be stopped. The algorithm was tested against static and moving obstacles achieving good handling performance while avoiding the obstacle.

A Dubin's curve approach was implemented as well for a parking maneuver in perpendicular and parallel parking [19]: 3 sub paths are calculated which define an initial turning radius. The turning radius circle is then dynamically updated depending on the relative position and heading angle of the vehicle.

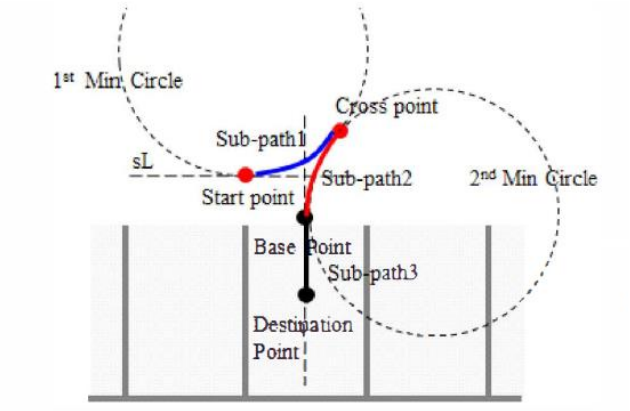

***Figure 2.7**: Three step parking maneuver for backward perpendicular parking [19]*

Bézier's curves where implemented in [15] using a three-step algorithm based on the Turning Standard Line: the path is obtained from unit paths starting with an initial known path to a final unit path in the parking spot.

The length of each unit path is limited by the collision-free space; three steps are present with initially a base candidate generation calculated from the start position. Secondly, a completion check is carried out which generates a path to the goal position from the previous step result: if this is not possible, a "rubbing path" step generates further paths to change the vehicle heading. This last step combines forward and backward unit paths.

The number of repetitions to find a feasible path is limited by a threshold value: finally, the path is optimized by minimizing total length and max derivative of curvature. This solution proved to be effective in narrow parking scenarios.

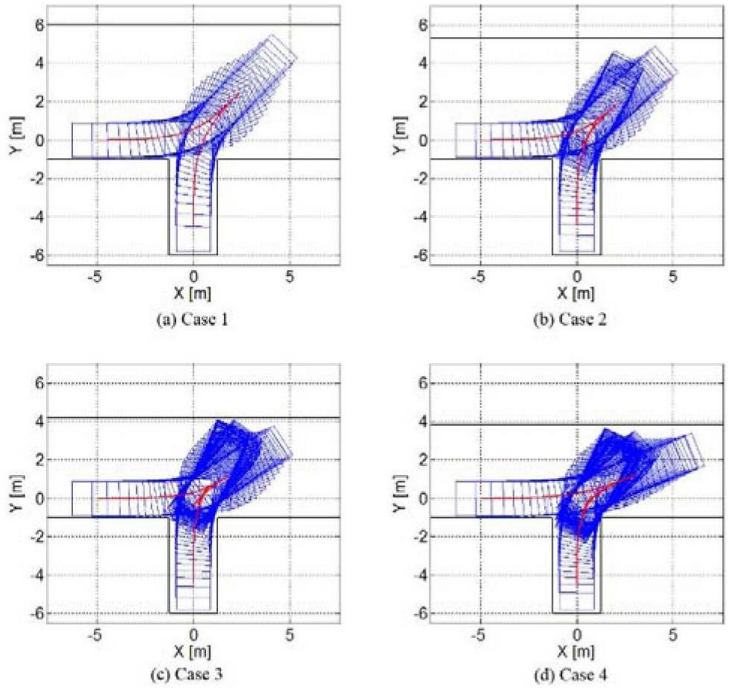

***Figure 2.8**: Parking path and final vehicle trajectory [15]*

Map based approaches to the path planning problem have also been successfully implemented.

In [14] a map is created by the vehicle on which an A* search algorithm runs in three stages: the first result is a series of edges the vehicle needs to cross during navigation, then a path smoothing is performed using a fourth order polar-polynomial function. A check of curvature is then performed and, if necessary, a further second step of smoothing is done.

Within this work a reactive path planning scheme was implemented to tackle any detected obstacles: this was needed as the electric vehicle has to park at a charging station with minimum offset and orientation errors.

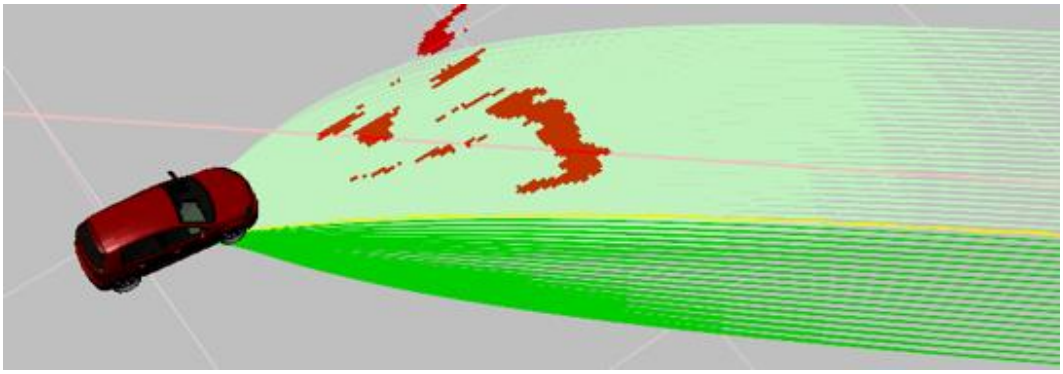

***Figure 2.9**: Set of obstacle avoiding trajectories (dark green) in a reactive path planner. [14]*

A second map-based approach was studied in [23], achieving state of the art performance by giving the vehicle information about the parking area.

The first step is to obtain a semantic layer from available geographical maps: this defines the drivable areas, center lane and the parking lot geometry. This information will be used as a reference in the path planning. The latter is a Hybrid A* algorithm which already takes into account Ackerman steering geometry specific to the vehicle and as such is capable of generating a continuous curvature path with no further optimization needed. This algorithm is then run on a metric layer made from a location-based grid map.

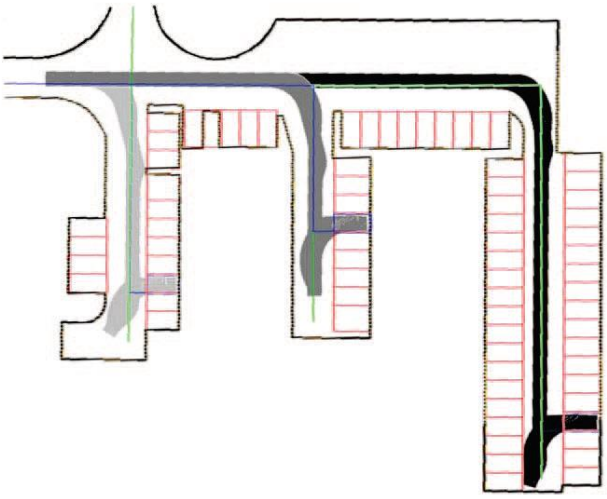

***Figure 2.10**: Three different parking paths planned by Hybrid A* [23]*

The results show that a minimum steering angle was used while closely following the reference path (light green in the picture). More interestingly, all three paths ended with a different final parking maneuver, showing great flexibility.

A modified Reeds-Shepp curve algorithm was developed in [24]. This path planner generated the shortest path while minimizing turning radius: the vehicle used DGPS and LiDAR to gather information on current position and parking lot geometry before executing the path planner.

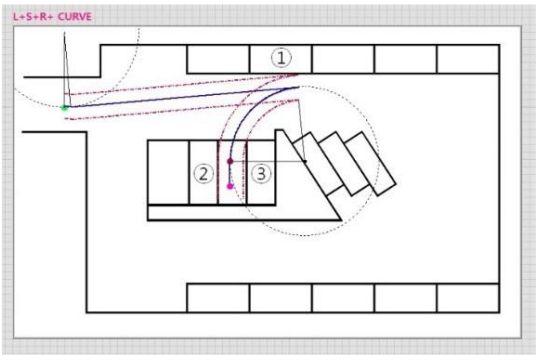

***Figure 2.11**: Backward perpendicular parking using Reeds-Shepp curves [24]*

In a peculiar application [32] a time optimized path planning algorithm is adopted for real-time trajectory generation: each constraint is provided analytically and then a dynamic optimization problem is solved. Being a direct method the resulting path was sensible to the constraints, which did not include null final heading angle to simplify the problem.

2.3 Vehicle Control

The last task in a AVP system is to automatically drive the vehicle along the desired path by providing lateral, forward and backward longitudinal control. This is achieved by equipping the car with electric actuators for steering, braking and throttle. The control is to be designed in order to have the vehicle track the planned path while minimizing tracking errors and input magnitudes.

The control would be designed based on the model adopted and will be constrained by characteristics of the driven vehicle such as steering kinematics and dynamics as well as brake pressure and vehicle acceleration. Robustness against disturbances like road slope or sensor measurement uncertainties must be ensured.

Owing to the low speeds of AVP and limited slip angles, the models can be simplified and kinematic models are widely used as in the following studies.

A bicycle model was used in [26] and compared to a more complete hatchback car model on CarSim®: a maximum deviation of 5° in the heading angle was found. A further validation against real data from an SUV car showed that a simple kinematic model is accurate enough in modeling low speed behaviour of a vehicle. In the same work, a nonlinear control technique, Dynamic Surface Control, was applied to design a lateral controller providing inherited robustness with respect to noise and time delay in position measurement [34] while considering the constraint on steering angle and steering angle rate. For a forward vehicle speed of 8km/h the lateral controller proved to be stable and simulation carried on real SUV car implementing an AVP system showed a max lateral error of about 0.23m for forward driving and 0.32m in backward perpendicular driving.

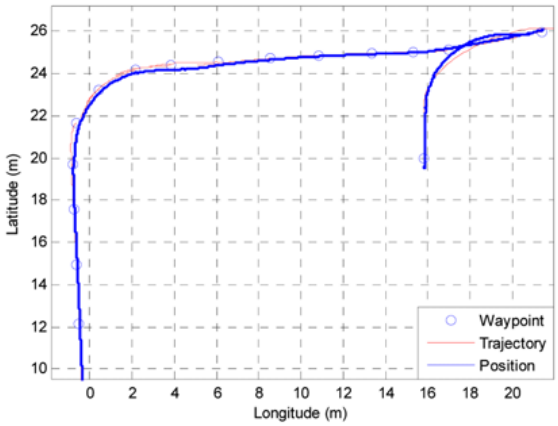

***Figure 2.12**: Tracking performance from field tests with an SUV car in [26]*

A closed loop fuzzy controller was implemented in [27] using as tracking strategy a distance-angle bias preview. Within this strategy the traverse distance and heading angle error and their derivative are provided as an input. The controller showed good tracking performance and robustness in initial MATLAB® simulations of backward parallel parking. Further validations were completed in PreScan® virtual scenarios with ultrasonic sensors equipped cars.

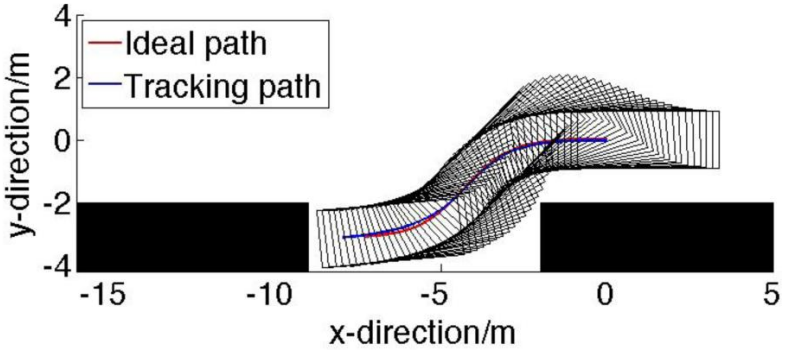

***Figure 2.13**: Tracking results of Fuzzy controller from [27]*

A control strategy for an electric vehicle was designed in [28] within a park assist architecture. Two different controllers were developed; a PI speed controller for the electric motor was included in a trajectory tracking controller controlling also the EPS module. Moreover, a simple kinematic model was employed, and the controller asymptotic stability, around the origin, was demonstrated with Lyapunov Function Method.Therefore, the controller was able to keep the vehicle on the reference trajectory with the presence of uncertainties and disturbances during both forward and backward driving. Performance was nonetheless limited by the simplification in deriving the tracking controller from the kinematic model.

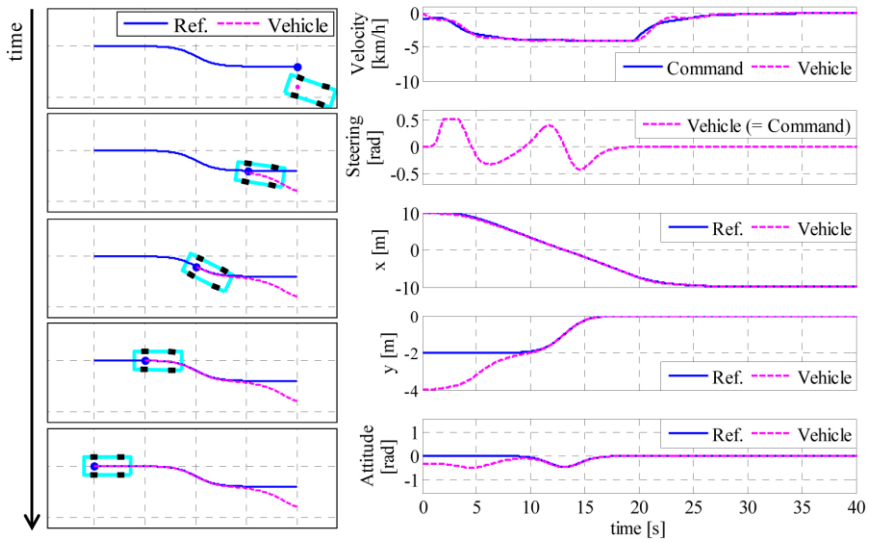

***Figure 2.14**: Simulation results of tracking control for the control strategy proposed in [28]*

An application focused primarily on optimizing the end of maneuver longitudinal control is found in the work [29]. The aim is to bring the vehicle to a stop with high accuracy while minimizing inputs from the actuators: road grade and engine idling behavior were both accounted for as disturbances through a disturbance estimator.

The controller is also able to track precisely backward low speed references and to bring the vehicle to a stop if obstacles are detected. These functionalities are made possible by using a hybrid controller, which included a wheel count distance estimator, an ultrasonic target generator for static obstacle tracking and a distance-velocity error estimator for the feedback control. The resulting controller would provide three modes of longitudinal control: speed limiting (low speed cruising), distance tracking (parking maneuver) and stand still (end of maneuver). The system was tested in different slope configurations and with unexpected obstacles, showing satisfying performance.

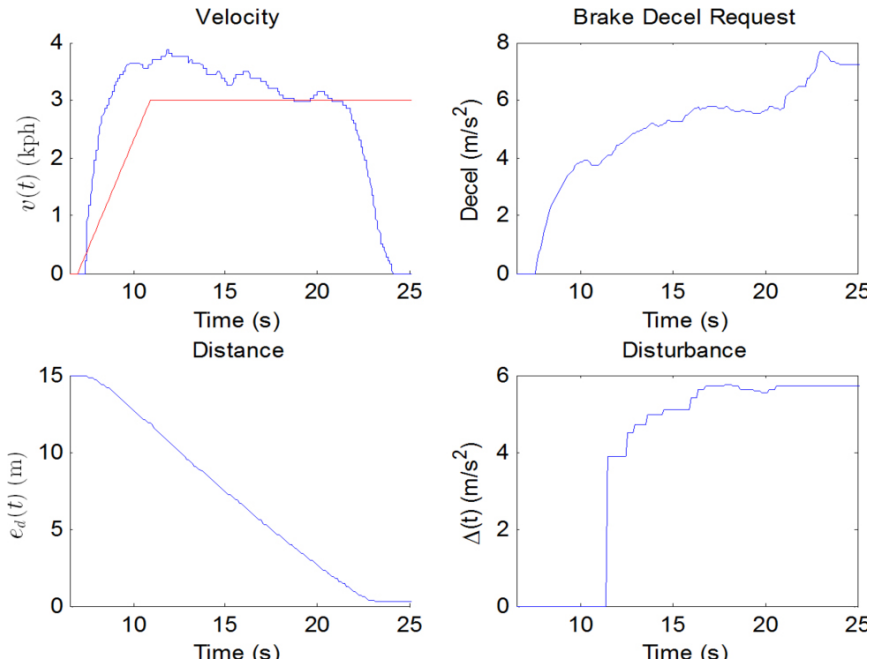

***Figure 2.15**: Experimental test results with transition from flat to sloped ground [29]*

LQR was employed in [30] to provide optimal feedback control; this was defined from the estimated lateral and yaw angle errors. Time varying parameters (loop gains and delays) were used to account for powertrain and external disturbances alike. The controller was implemented in a LiDAR equipped vehicle which performed an automatic valet parking maneuver with an overall path tracking error of less than 30cm; yet an overshoot of around 1m occurred when switching to reverse, due to reference path changing.

A peculiar policy-based control solution is presented in [39] where local feedback policies are defined in logical terms to guide an automated vehicle in accomplishing the high level command “drive around until you find and empty parking space, then park”.

3 MPC theory and application

In the following chapter a brief theoretical introduction about model predictive control and its application is given.

Model predictive control is an optimization-based control strategy in which a model of the controlled system is used to predict its future states. From this prediction, an optimal control input that satisfies all system constraints is then calculated.

It is a widely used control strategy in industry, especially in process control applications. Due to its effectiveness and flexibility, MPC has found large application in automotive controls as well, as in [41] [42] [43].

The advantage of MPC is the ability of handling non-linear MIMO systems in which different parameters and variables interact between each other in complex and unpredictable ways. The information about the controlled system is embedded in the controller it self: this includes also the physical and variable constraints, which allows the solution of an optimal control.

One key aspect that largely affects the performance is the availability of a model of the controlled plant: this model has to be as accurate as possible in order to produce accurate states' predictions. These accurate models can be obtained through system identification [44] or by simple linearization of the system around a working point. It will be then of paramount importance to properly define this operating point as the optimality of the control will be strongly affected by it.

An overly complex predictor model, on the other hand can yield a complex optimization problem that may hinder real-time performance of the controller. However, recently the increase of processing power of microprocessors has allowed the adoption of MPC also in those applications where a time-sensitive control strategy is needed, such as aerospace, robotics and automotive applications.

A common application of MPC is when the system is needed to track a given reference: in the AVP case, the reference is the planned trajectory. Due to its predictive nature, the controller in this case considers not only the current reference tracking error, as in a usual feedback control, but also a future estimate after a given number of time

steps in the future. This prediction window is what is defined as the prediction horizon of MPC. It represents the number of control intervals that the MPC controller estimates by prediction when optimizing the control.

Deciding on a proper prediction horizon length is key to the success of the control; this is usually a function of the process sample time and the particular application. The sample time sets the rate at which the optimization is run in the control algorithm: a small sample time allows faster reaction to disturbances at the expense of larger computation load.

The recommendation is usually to choose a sample time between $\frac{T_r}{10}$ and $\frac{T_r}{20}$ with T_r being the rise time of the open loop system response. In the present work the sample time was chosen to be 0.01s as this is the fixed sample time required by the vehicle dynamics' S-Functions.

The prediction horizon p is usually chosen in the early phase of control design and it is gradually increased until internal stability of the closed-loop system is reached or if no further improvement in the controller performance is obtained: when the prediction horizon is increased beyond this the control problem poses a computational challenge. A trade-off is then needed. The recommendation here is to set a prediction horizon of 10 to 20 sample time steps within the open loop response time but now defined as setting time, that is the time needed to obtain an output with a given error range from the reference.

The product between the sample time T_s and the prediction horizon p defines the response interval of the system that characterizes the closed-loop dynamics of the system, that is how fast the controlled output tracks the desired reference.

When a dynamic trajectory is employed, the prediction horizon may be constrained by the information range: the tracking error can be estimated to the furthest available trajectory waypoint. For proximity sensors, this can be limited to 8 meters.

A simple calculation yields that the prediction horizon upper limit would be higher than 300 when considering a sample time of 0.01s and parking speeds.

Another parameter used in MPC setting is the control horizon: this indicates the time steps for which the optimized control input is enacted by the controller. All other optimized control inputs are discarded at every iteration of the control and new ones are calculated instead: the literature refers to this as the “receding control horizon”. Usually a control horizon of few time steps is sufficient as only the initial few control inputs have the bigger effect on the controlled output. Again, the control horizon may be set from $0.1p$ to $0.2p$.

In MPC, the control horizon is usually less than the prediction horizon, which is intuitive to understand: one cannot calculate an optimal control on states that have not been estimated yet. Properly tuning the control horizon is not a straightforward task: a too large control horizon improves the control performance on one side, while increasing the computational complexity on the other. Moreover, a small control horizon may improve the controller stability yet provides a less than optimal control.

3.1 Traditional MPC

In the present work, MPC is applied to the lateral control of a vehicle driven on a predefined trajectory. This trajectory is the reference that the controlled system needs to track in order to meet the performance requirement.

In the following, the working principle of this MPC approach is explained.

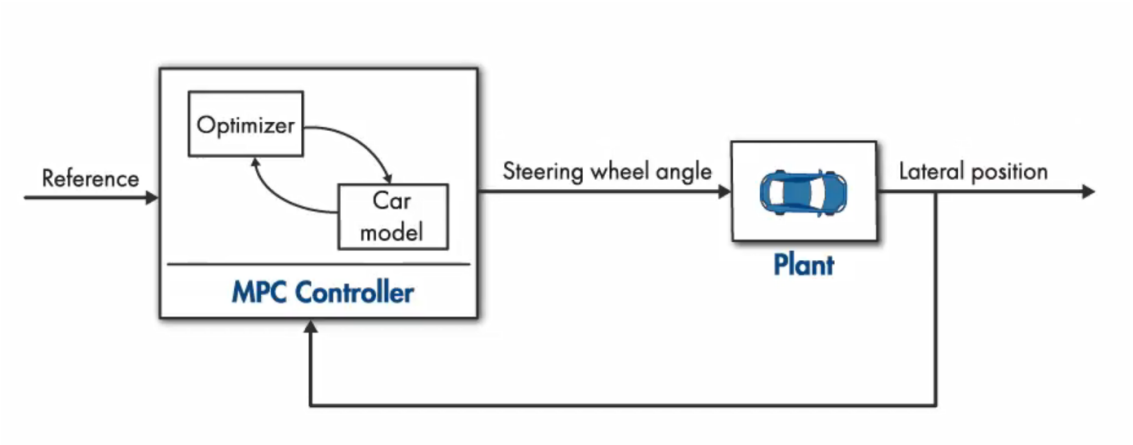

***Figure 3.1**: MPC control structure example [49]*

The car model is used to simulate the lateral position of the car in the next p time steps as a function of estimated control input that is the incremental steering wheel angles. The estimated control input is obtained through an online optimization where the objective is to minimize the predicted error while not exceeding a control input magnitude. This can be constrained for different reasons: physical limitations of the electric steering actuator or comfort requirements in terms of lateral vehicle acceleration.

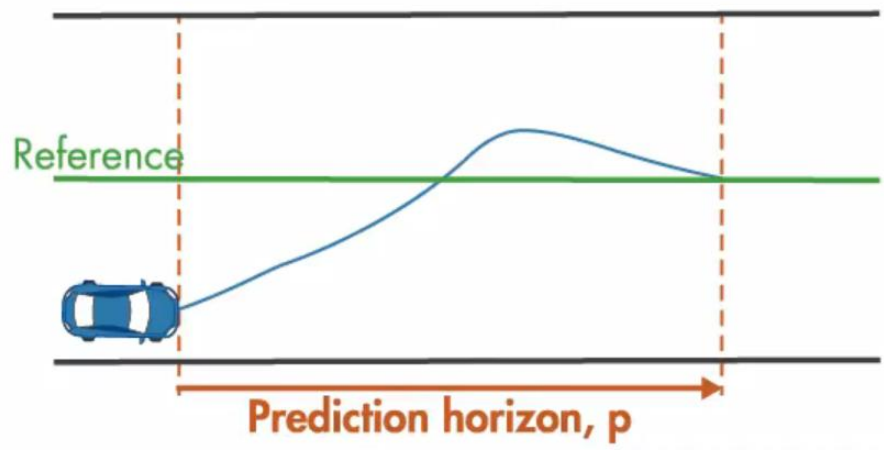

***Figure 3.2**: Representation of MPC application in trajectory tracking [49]*

In the literature, the optimization is commonly achieved by minimizing a cost function:

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

The problem at hand is optimized when both the predicted input, defined as a sum of incremental inputs Δu, and the prediction error e are minimized over the prediction horizon p. In the equation above the cost function to be minimized is defined as a sum of two quadratic terms in which the prediction error and input are multiplied by relative weights: these can be tuned during the design stage to obtain the best trade-off between minimizing the steering input and the trajectory tracking error. For the present application we require a tracking error, that is the vehicle center lateral position error relative to the trajectory to be less then 0.1m.

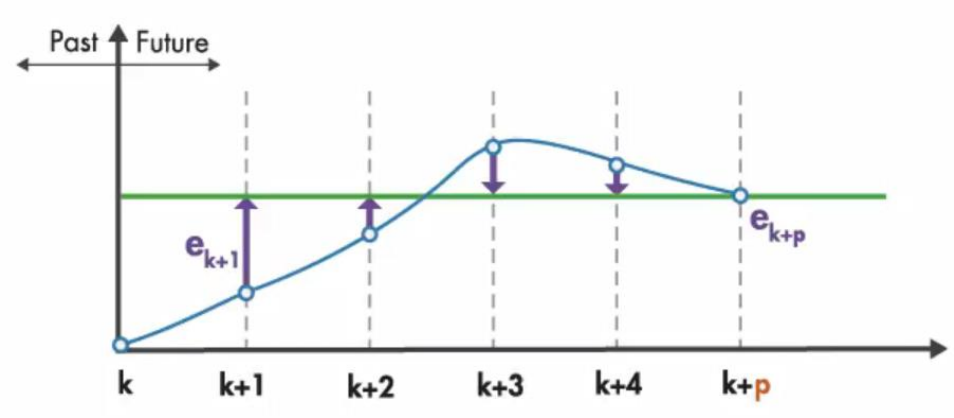

***Figure 3.3**: Tracking error by time steps within the prediction horizon [49]*

3.1.1 Constraints & Weights in MPC

First let us introduce some nomenclature: the control variable, that is the quantity that is used to enact the control (e.g. steering torque) is usually called the manipulated variable while the output variable (e.g. vehicle center offset from trajectory or q) is usually called measured output.

In MPC design, several important parameters need to be set in order to properly tune the controller to the plant's requirements. One of these are the already mentioned constraints, which can be applied to both the control variable and the output variable.

Constraints can be set on the variables and their rates as well; one can define hard constraints and soft constraints. Hard constraints cannot be violated and usually stem from physical limitation of the system (e.g. throttle input), while soft constraints may be temporally violated.

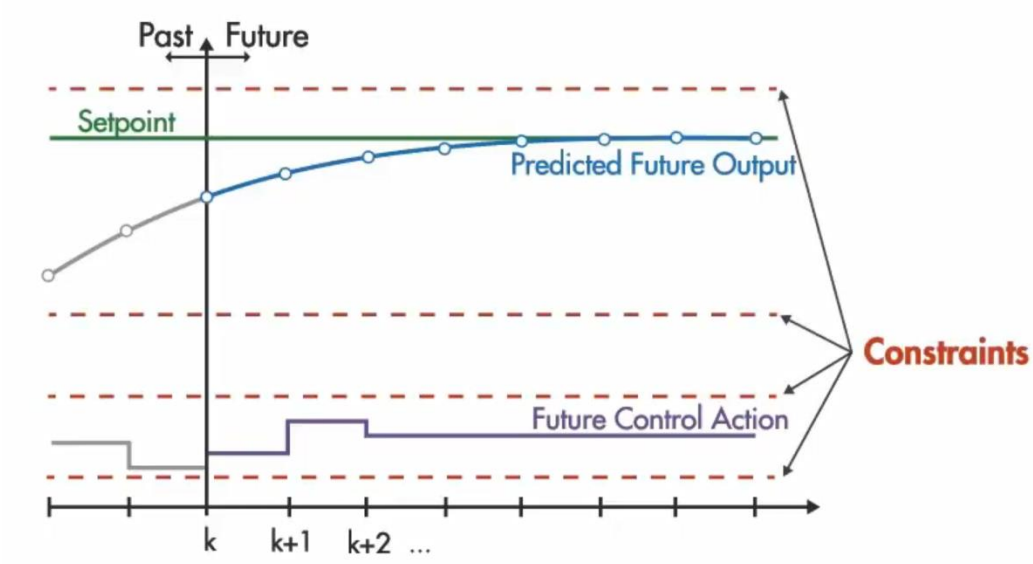

***Figure 3.4**: Constraints definition in MPC variables [49]*

When during the design several constraints are being inserted, difficulties may arise: the constraints introduced in some cases generate conflicts which may lead to unfeasible control optimization. Especially when using hard constraints, one should avoid introducing them for manipulated variables or their rate of variation.

Another important parameter that are commonly used are weights. These are introduced to better tune the controller behavior and to mitigate the impact of the constraint. In the aforementioned cost function, weights are chosen with the aim to balance the controller between having a more or less aggressive control action or reaching a more or less precise reference tracking. Weights tuning is key to obtain the desired controlled system dynamics, particularly in MIMO systems

3.2 Advanced Model Predictive Control

Complex systems and high control performance requirement may pose a challenging problem to solve from an MPC application perspective; to this end several evolutions of MPC controllers have been developed to meet the needs of control design.

The optimization problem that is defined from an MPC controller is usually solvable whenever the following conditions are present:

- *Linear system*
- *Linear constraint*  *Linear Time Invariant MPC*
- *Quadratic cost function*

With these conditions met, the optimization problem becomes a convex one that is for which a local minimum of the cost function to be me minimized is guaranteed to be a global one, thus yielding an optimum solution.

However, the above ideal hypothesis may not be satisfied and a different approach to MPC applications is then needed: in the following, some of these approaches are briefly presented.

When the system to be controlled is a nonlinear system, MPC can adopt linearization to simplify the problem.

3.2.1 Adaptive MPC

In this implementation of MPC, the nonlinearity of the controlled plant is dealt with using a step by step linearization approach. As the dynamics of the system may be changing significantly from one instant to the other, the model that the controller employs to predict the states is updated accordingly for every set time step.

Adaptive MPC is particularly suited for applications where the system has great dynamic variety within the control space of interest and using a single identified model or a linearized model would yield poor control performance or robustness.

In this declination of MPC, the number of states and constraints are fixed: for those applications in which flexible constraints are needed, then a Gain-scheduled MPC is adopted. Several models are obtained offline at different operating points and each one is embedded in a different controller: a switching algorithm will then choose the right MPC controller depending of the operating condition. These models can be of different number of states and the relative controllers can use different constraints and cost functions.

3.2.2 Explicit MPC

As already stated the MPC controller task is to solve an online optimization problem; the complexity of this problem grows considerably with increasing number of states, constraints, prediction and control horizon sizes. For those applications in which fast sampling is present along with demanding dynamics, the need for real time control performance creates a great hurdle on the on-board hardware.

Therefore, a different MPC approach can be employed: explicit MPC. When the state space of the controlled plant can be mapped one can precompute optimized solutions for properly chosen regions of the state space. These regions are determined by ranges on the states which define functions that are linear and piecewise continuous. In the figure a single space example is proposed, where the x axis domain is divided in different intervals and $fi + gi$ representing the local linear functions: the overall explicit solution $u(x)$ is then obtained by combining the local ones. The optimization problem becomes then an evaluation of offline precomputed solutions, which greatly improves the runtime efficiency of the control algorithm.

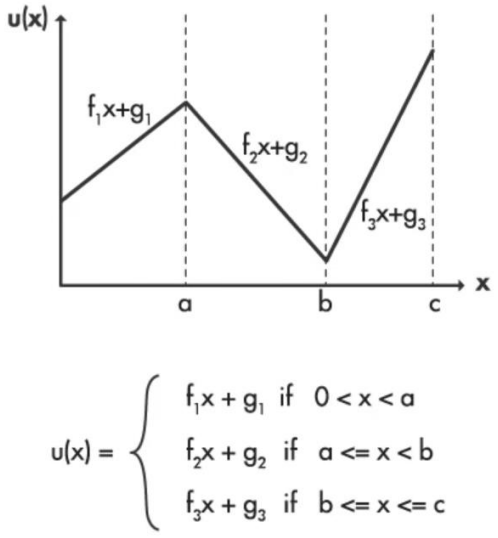

***Figure 3.5**: Definition of solution regions in a single space system*

4 Model components

In the present chapter a description of the models used in the simulation and their definition is provided. The simulation software employed, in particular the Simulink MPC toolbox, is also presented.

4.1 Simulation software

For the present work the main software of reference was MATLAB® 2012-32bit and in particular Simulink was used to set up the simulation in its graphical programming environment. This seemed convenient, as many of the simulation model components were already made available from the OEM as Simulink blocks.

Within this release of Simulink® the Model Predictive Control Designer Toolbox® was especially useful in setting and properly tuning the MPC controller to the design requirements. For the sake of clarity, the toolbox will be further discussed when dealing with simulation setting in the next chapter; now a close look to the key components of the simulation is given.

4.2 Electric Power Steering

Given that an automated vehicle is steered by controlling an electric actuator acting on the steering column, integral with the EPS module, it dictates that a model of the EPS system needs to be included in the simulations. Again, the system was obtained thru advanced system identification and it is used as provided.

While the EPS function is to assist the driver inputs by providing torque adjustments, for comfort or safety reasons, here the EPS is modelled as a “black-box” transfer functions that translate the two listed controlled inputs into a corresponding output:

-steering torque [Nm]
-vehicle speed [km/h]

Internally the system is defined as Linear Time Invariant (LTI) system and when analyzing its output, its behavior can be considered as a small delay low-pass filter with regard to the steering torque to angle response, as the graphs below demonstrate.

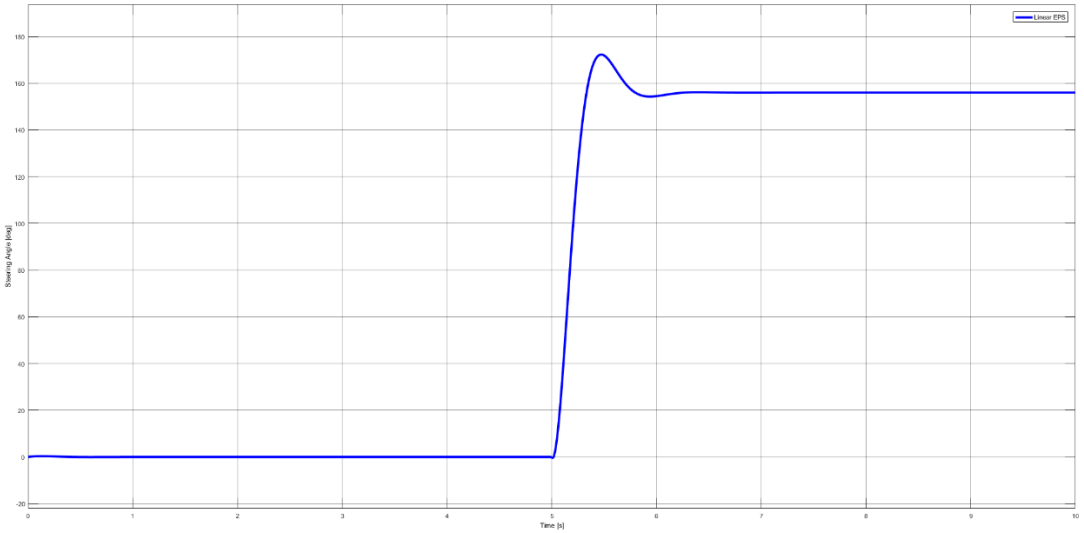

Figure 4.1*: EPS Steering Angle [deg] response for a 7 [Nm] Steering Torque step input at 5s at a vehicle speed of 8km/h*

From the plot above we notice a slight overshoot in the steering angle step response which can be explained by the damping (0.58) provided by two negative real part conjugate complex poles defining the dynamics of the system.

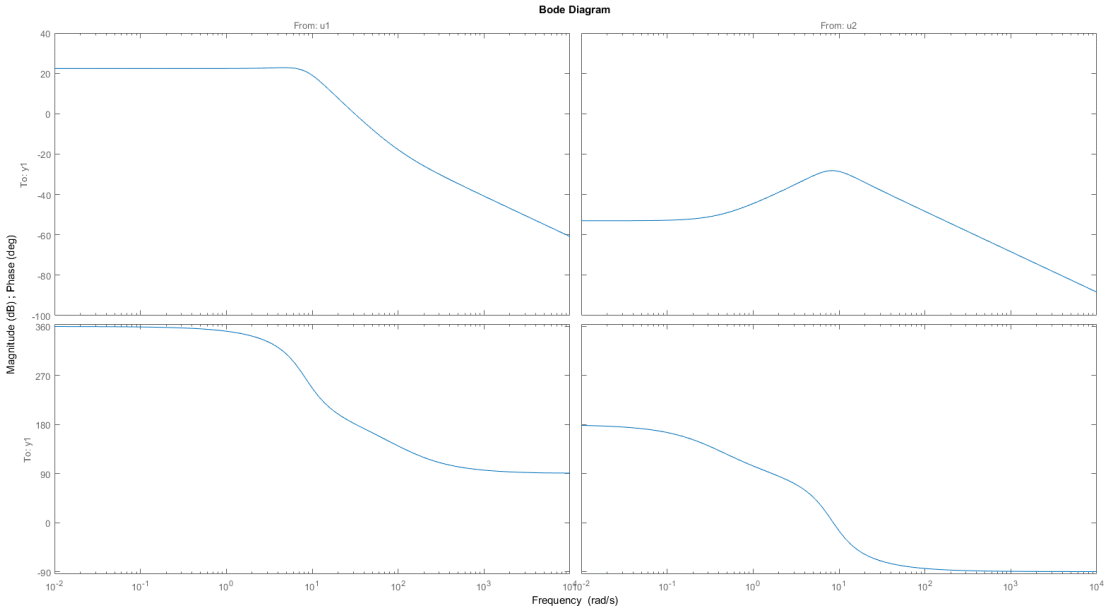

***Figure 4.2**: EPS system Bode plot*

The first Bode plot (on the left) shows how the system manages to filter the high frequency torque signals, as the ones generated by the road profile, while providing a stable and predictable gain for the usable frequency range of the steering. In the second Bode plot, the transfer function gain between the longitudinal velocity and the angle is shown: as already noticed when evaluating the system response, the effect of vehicle speed on system behavior is fairly limited. In general, speed variations are limited due to intrinsically small decelerations present in parking environment as such the system will not be affected by its second input.

4.3 Vehicle Dynamics

In order to simulate the vehicle control, a proper representation of the vehicle behavior needs to be provided. To this end, two S-functions for both the vehicle longitudinal and lateral dynamics were used. These functions have been defined starting from experimental data gathered from the field drive tests.

The definition of the functions themselves are not accessible, as such they are dealt with as “black-box” models having only regard towards the selection of proper inputs and outputs.

Owing to the limited speeds in parking scenarios in many applications presented in the literature review a series of simplified or low-fidelity models are used as at low speeds the lateral behaviour of the vehicle approaches that of the kinematic model. The reason behind this is the limited side-slip at the tyre level [45] which allows the hypothesis that only negligible nonlinearities are introduced during the vehicle maneuvering: indeed, the true advantage would be the great computational simplification of the problem. A good trade off could be ideally to identify a vehicle model, both lateral and longitudinal, from a real vehicle driving in a parking space. This vehicle model has to account for several disturbances in which effects are usually amplified at low speeds, namely the road slope and its variations and powertrain induced disturbances (e.g. engine idling, driveline vibrations).

4.3.1 Lateral Dynamics

The lateral dynamics Simulink block models the lateral behavior of the vehicle. Taking as inputs the steering angle [rad] from the EPS model and the longitudinal speed V_x [m/s] from the longitudinal one, the lateral dynamics block produces then as outputs:

- $\dot{\Psi}$ [rad/s] Yaw Rate
- V_y [m/s] Lateral Vehicle Velocity
- a_y [m/s^2] Lateral Acceleration

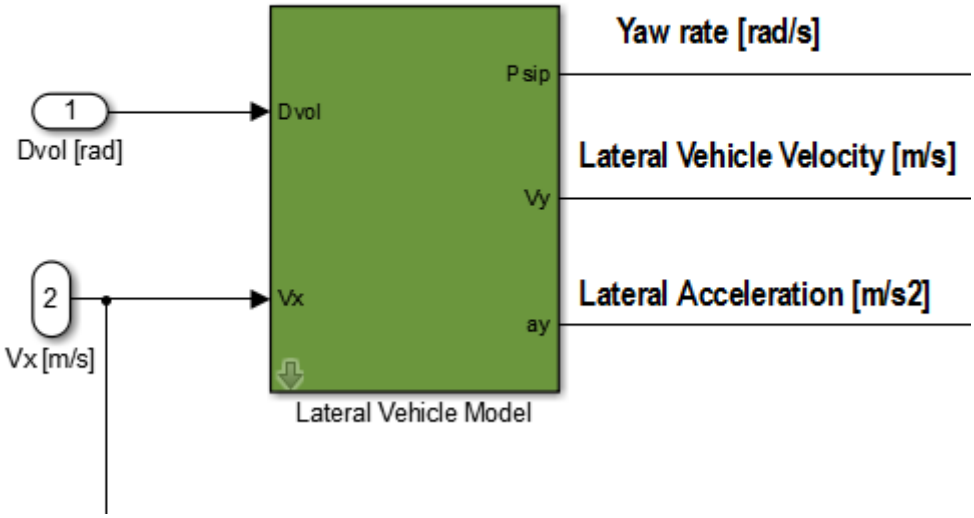

***Figure 4.3**: Lateral Vehicle Dynamics model in Simulink with I/O definition*

These outputs are then used to define the LRC block dynamics and the coordinate transformations that allows reconstruction of the vehicle position relative to the desired path.

When designing the control strategy, it was found that the lateral dynamics block was highly sensitive to the longitudinal speeds value. This can be provided as a constant low speed reference, but it was noted how large instabilities were triggered when this speed approached a null value. Indeed, this mimics the output of a “bicycle-model” like block, where the longitudinal speed is present at the denominator, as it can be seen in the state space model definition in the next page.

$$\frac{d}{dt}\begin{Bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{Bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\frac{2C_{\alpha f}+2C_{\alpha r}}{mV_x} & 0 & -V_x-\frac{2C_{\alpha f}\ell_f-2C_{\alpha r}\ell_r}{mV_x} \\ 0 & 0 & 0 & 1 \\ 0 & -\frac{2\ell_f C_{\alpha f}-2\ell_r C_{\alpha r}}{I_z V_x} & 0 & -\frac{2\ell_f{}^2 C_{\alpha f}+2\ell_r{}^2 C_{\alpha r}}{I_z V_x} \end{bmatrix} + \begin{Bmatrix} 0 \\ \frac{2C_{\alpha f}}{m} \\ 0 \\ \frac{2\ell_f C_{\alpha f}}{I_z} \end{Bmatrix} \delta \quad (2)$$

***Figure 4.4**: State-Space Lateral Vehicle Dynamics model in Simulink with I/O definition [45]*

The state space definition of the lateral dynamic model uses the lateral vehicle position y, lateral speed $\dot{y}$, yaw angle Ψ and yaw rate $\dot{\Psi}$ as state variables; other parameters used include tire cornering stiffnesses C, center of gravity to axles distances l, vehicle longitudinal speed V_x, vehicle mass m and vehicle inertia around the vertical axis I_z. The front wheels steering angle δ is present as an input

In the simulation setting, this instability was dealt with primarily by two means: first a non-null initial vehicle speed is defined ($V_{Xinitial}$ > 1 m/s) and then a saturation block is included in the longitudinal speed output as to avoid feeding close to zero longitudinal vehicle speeds to the lateral dynamics block.

4.3.2 Longitudinal Dynamics

The second S-Function employed is defining the longitudinal dynamics of the controlled vehicle having two physical quantities that need to be specified in the block: one is the already discussed initial vehicle speed, which is chosen to be 1.5 m/s, while the other one is the engine friction torque that was not specified.

Regarding the definition of the inputs, there is a total of five that can be provided:

- Initial Gear Inserted
- Gas Pedal [%]
- Master Cylinder Pressure [%]
- Wind speed [m/s]
- Road slope [rad]

The last two of these inputs are discarded as they were not relevant for the application or, as in the case of road slope, they were not specified. Indeed, road slope is an important element of disturbance that a robust control system design should take into account: in this application its effect is greatly amplified by the low speeds.

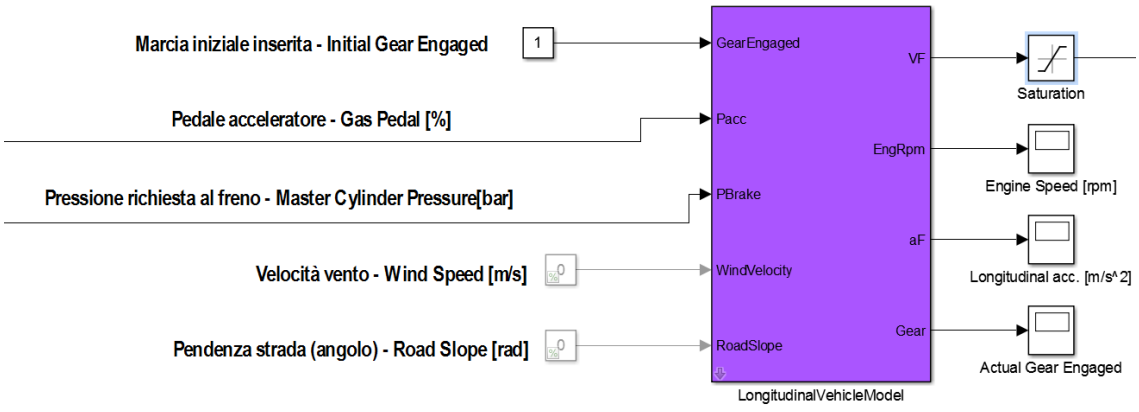

***Figure 4.5**: Longitudinal Vehicle Dynamics model in Simulink with I/O definition*

Of the four outputs generated only the longitudinal speed being used for both the lateral dynamics and the feedback PID vehicle speed controller, after proper unit conversions.

4.4 Lane Recognition Camera

The dynamics models described in the previous sections allow to develop a model for an LRC sensor that provides the information needed to control the vehicle.

The scheme is widely discussed in the literature [46] and easily applicable; four inputs are needed to define the system which are the following:

-V_x $\left[\frac{m}{s}\right]$ Longitudinal speed
- V_y $\left[\frac{m}{s}\right]$ Lateral speed
-$\dot{\psi}$ $\left[\frac{rad}{s}\right]$ Yaw rate
-K_l $\left[\frac{1}{m}\right]$ Road curvature

With the quantities above the LRC model provides information about the position of the vehicle centre of gravity relative to the road centerline approximation a distance q along the y axis, and the angle between the vehicle longitudinal axis direction and the road centerline approximation ($m[rad]$). Both these quantities are assumed to be zero initially.

In the present application the centerline information is actually the planned trajectory that is translated in terms of path curvature.

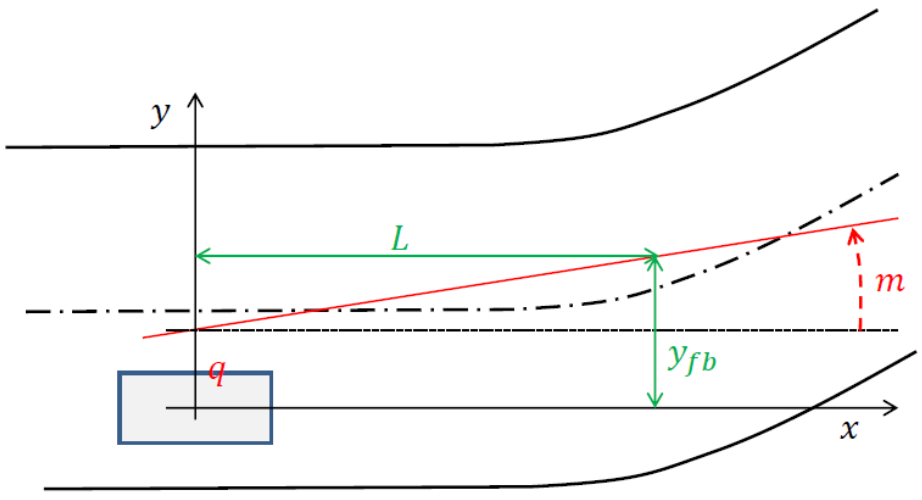

***Figure 4.6**: Lane Recognition Camera model representation*

The equations of the system are the following:

$$\dot{q} = V_x m - V_y - L V_x K_l \qquad (3)$$

$$\dot{m} = V_x K_l - \dot{\psi} \qquad (4)$$

Here an important parameter is introduced: $L[m]$ that is defined as the "look-ahead distance". The use of this parameter allows to evaluate the offset of the vehicle center of gravity at a distance ahead on the road: with this information we can more effectively control the vehicle as this approach mimics that of a human driver when he or she is steering a vehicle. [47]

Using this parameter, we evaluate $y_{fb} = mL + q$ which predicts the position of the vehicle at the distance L ahead: properly tuning this parameter is key to the performance of the control. In traditional applications of Lane Recognition Systems, as in a Lane Keeping System, when a too large L parameter is adopted the vehicle has a tendency to cut thru corners and be more reluctant to direction changes. On the other hand, a too short L distance induces a more erratic motion of the vehicle. Needless to say, these effects are magnified at higher speeds.

When considering a feedback control of the vehicle it would be more intuitive to consider the look-ahead vehicle y_{fb} position as the quantity of choice; however, this is not the case as the prediction capability of MPC allows the use of q as feedback quantity. This is understandable as the system evaluates the error prediction for future values of q, which is controlled by the prediction horizon. In other words, the parameter L is indeed present in MPC but as a product of three other parameters:

-prediction horizon p

-sample time T_s

-vehicle longitudinal speed V_x

Considering that the vehicle speed is limited in this application, from to 2 to 3 m/s in most scenarios, and the sample time T_s as already discussed is fixed, this leaves the prediction horizon as the only tunable parameter: as it will be shown in the next chapter this greatly impacts the controller effectiveness.

5 Simulation Setting

In this chapter, a description of all the phases involved in the controller design will be described, with particular focus on the use of MPC Designer Toolbox in MATLAB®.

The building blocks already described were assembled together in a single Simulink simulation: the next step was to generate the path information.

5.1 Scenario and trajectory definition

Initially a scenario design toolbox was used to define the parking environment. This included a limited drivable area (grey), a series of parked vehicles and barriers: the goal would have been to feed this scenario to the path planner but this capability being non-available a custom waypoint trajectory was instead generated.

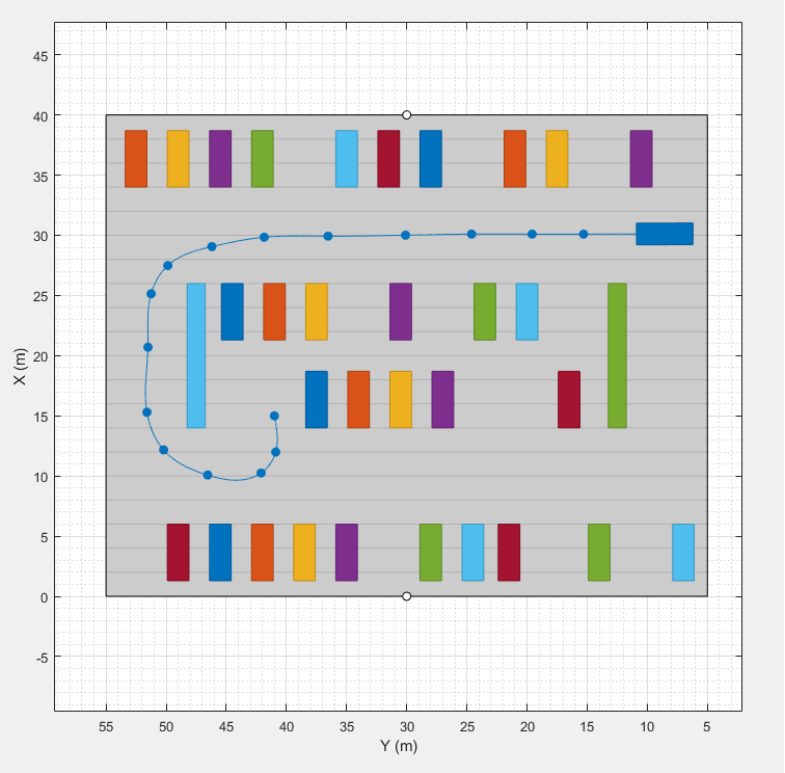

***Figure 5.1**: Parking scenario in Driving Scenario Designer®*

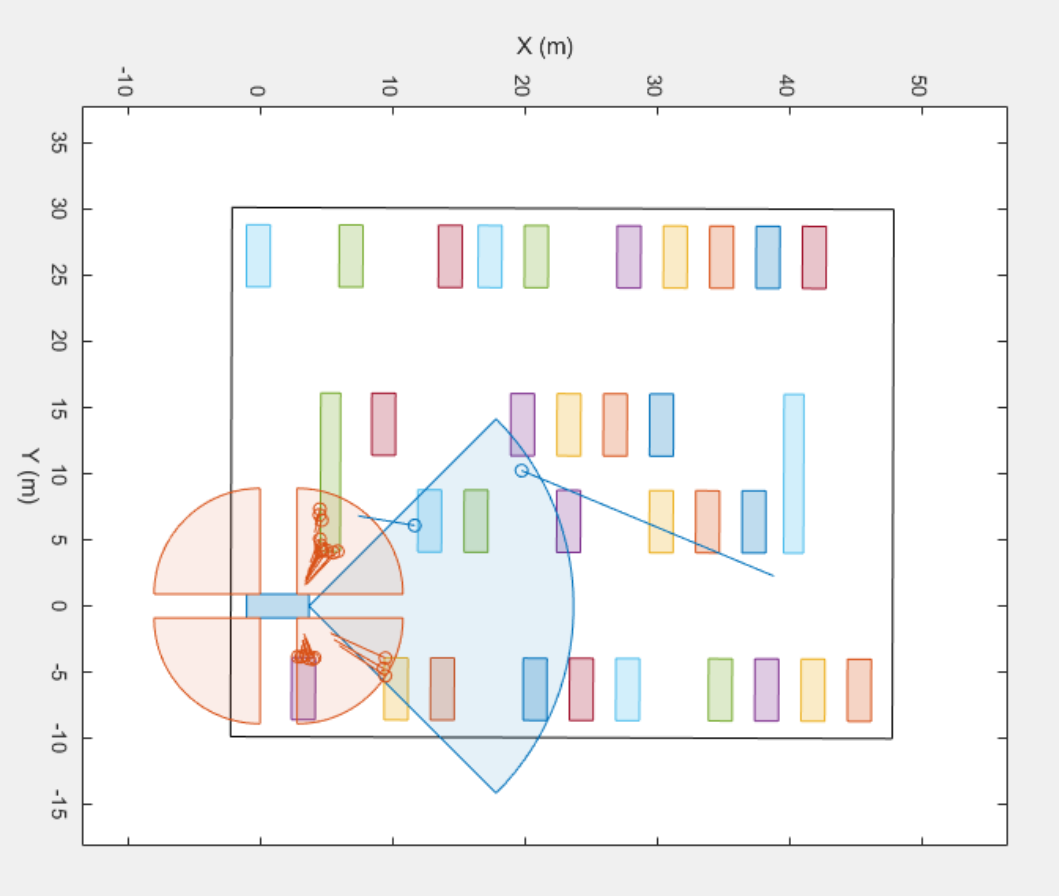

***Figure 5.2**: Bird's Eye view of the designed scenario: in blue Vision sensor range, in orange radar detection field*

The trajectory is then defined as a series of waypoints which are used to generate a full set of x and y points thru spline interpolation in MATLAB®. This command also allows the definition of initial and final slopes of the path, which were set to zero.

Once the path is defined in x and y coordinates, the velocity profile and curvature can be defined thorough simple derivations: as it is convenient, a time discretization of the length of the sample time was used for the derivations.

The following equations are used to define velocity and curvature:

$$V_x = \sqrt{\dot{X}^2 + \dot{Y}^2} \qquad (5)$$

$$K_l = \frac{\ddot{Y}\dot{X} - \dot{Y}\ddot{X}}{\left(\dot{X}^2 + \dot{Y}^2\right)^{3/2}} \qquad (6)$$

5.2 Longitudinal Controller

Once the road curvature and velocity profile are defined these can be provided to the controllers. For the longitudinal controller, the goal is to track a low speed reference by actuating the accelerator or the brake accordingly: to this end a simple PID controller is used. The controller gains were tuned by trial and error until a satisfactory speed tracking was achieved.

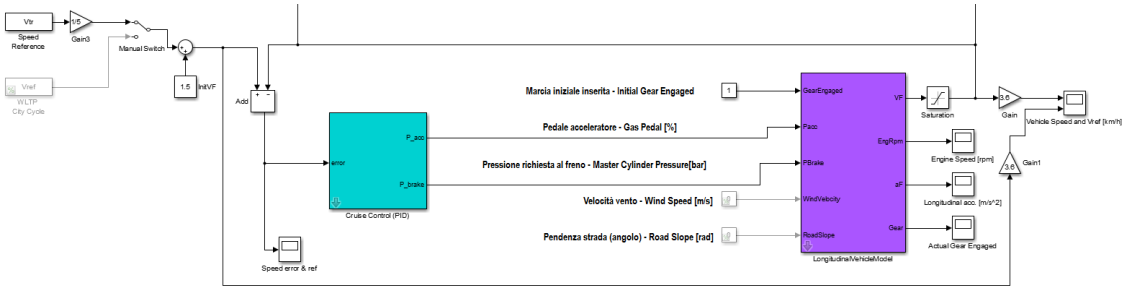

***Figure 5.3**: Longitudinal Dynamics PID Controller (Cruise Control)*

In this application only a PI controller was sufficient as the vehicle accelerations are limited, as such no derivative term is needed: when examining the gains, it can be seen that a high proportional gain alone allows precise tracking of a reference speed ranging between 12 and 15 km/h, representative of vehicle speeds in parking scenarios.

5.3 Lateral MPC Controller

The design of the lateral controller was the next step to as a reliable and robust longitudinal speed tracking is now available. After a presentation of the MPC Designer app a step by step description of the controller design is given in the following paragraphs.

5.3.1 MPC Designer Toolbox

This toolbox presents it self as an effective and easy to use tool that guides the control engineer in properly designing his controller: not only is it possible to simulate the controller for custom scenarios but it is also possible to study the stability of the controller and of the closed-loop system with a in built design report.

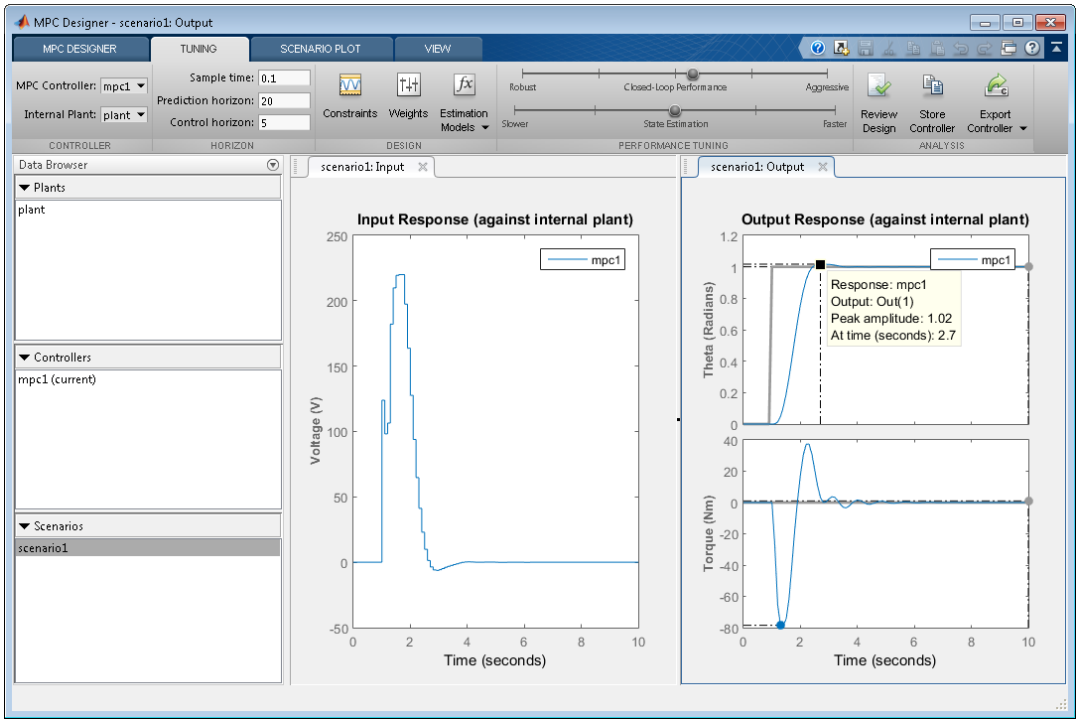

***Figure 5.4**: MPC Designer Toolbox® environment*

Initially the MPC structure has to be defined in terms of system linearization and I/O setting; second a tuning phase follows, in which horizons, weights and constraints are evaluated. Within this tab also the Review Design function can be used to study of the system before running the various simulations.

5.3.2 Model I/O Definition

As already mentioned the definition of input-output structure greatly impacts the performance of the controller. The lateral controller task is that of providing a feedback control through the steering torque T_{dr} $[Nm]$ to allow the vehicle to track the trajectory: this will naturally the be choice of the manipulated variable. On the input side more evaluation is needed before deciding the inputs of the controller.

The MPC controller function is that of tracking a given reference while in presence of a certain disturbance deviating the vehicle from its path. When the state chosen for the feedback is the q position of the center of mass of the vehicle, instead of future position y_{fb}, it becomes clear that providing a trajectory reference ($q = 0$) to track will greatly simplify the problem. The controller will then calculate the optimal torque to steer the vehicle in order to obtain a close to zero q value.

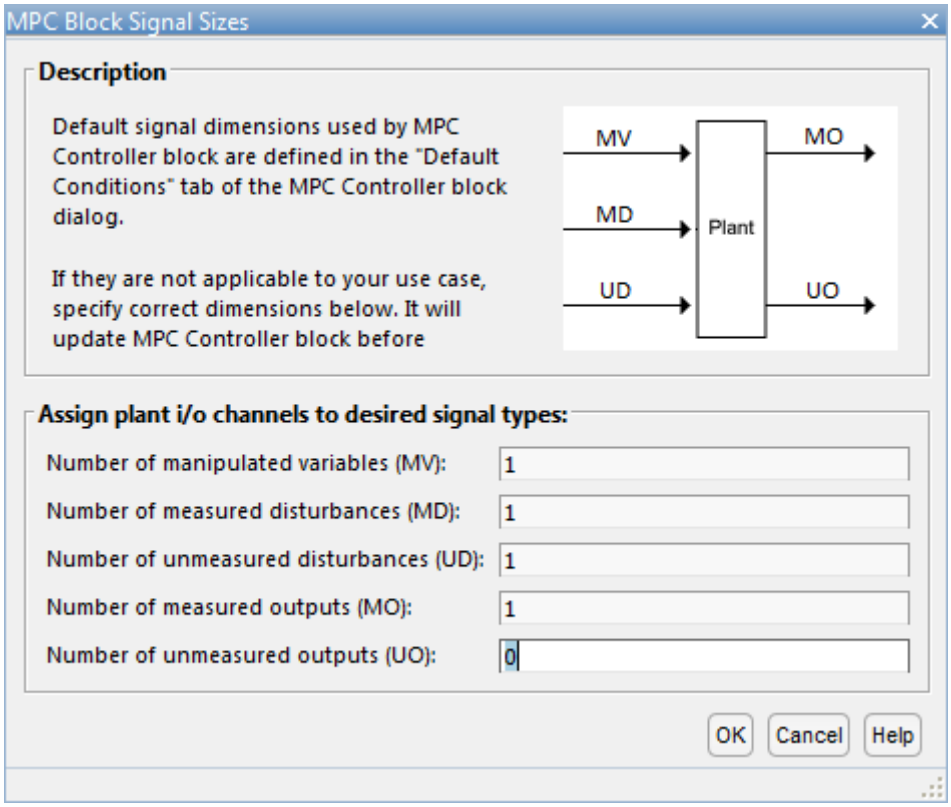

***Figure 5.5**: MPC Controller I/O definition in MPC Designer Toolbox®*

Along with q, the controller receives as input also the road curvature: this is because an MPC block with measured disturbance was used. Using this approach, a more robust control is obtained as compared to a system in which the road curvature is only present within the lateral dynamics of the system. While the linearized model, described in the next paragraph, still includes the road curvature as input to the system, the controller greatly benefits from having the road curvature as a separate input. When considering the AVP architecture as a whole, it is understandable how this information may well be available in real time to the controller: even in applications where advanced path planning is used such as dynamic planners for obstacle avoidance this information can be still made available for the MPC controller.

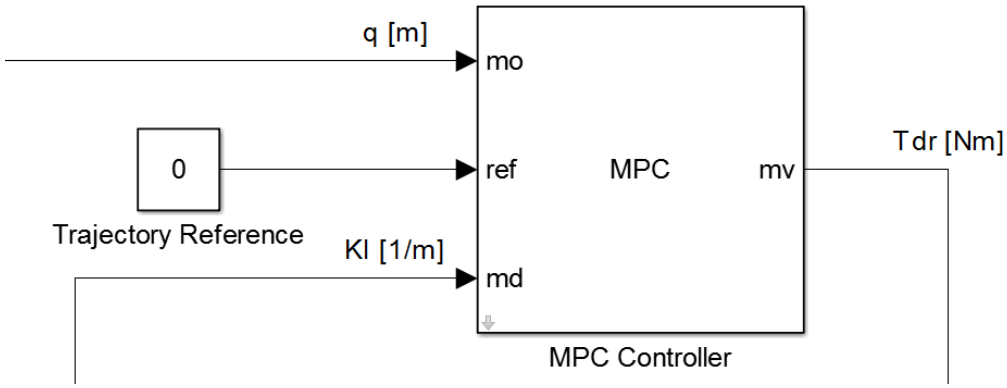

***Figure 5.6**: MPC Controller I/O definition in Simulink*

5.3.3 Model linearization

Once the inputs and outputs of the system model have been defined, the linearization can proceed to obtain the embedded linear predictor model. In order to linearize, an operating point of the open loop system must be chosen: this has a great effect of the remaining design of the controller.

The operating point was chosen at 15s in the simulation of the uncontrolled system as to avoid initial disturbances observed in the states, especially due to the EPS model.

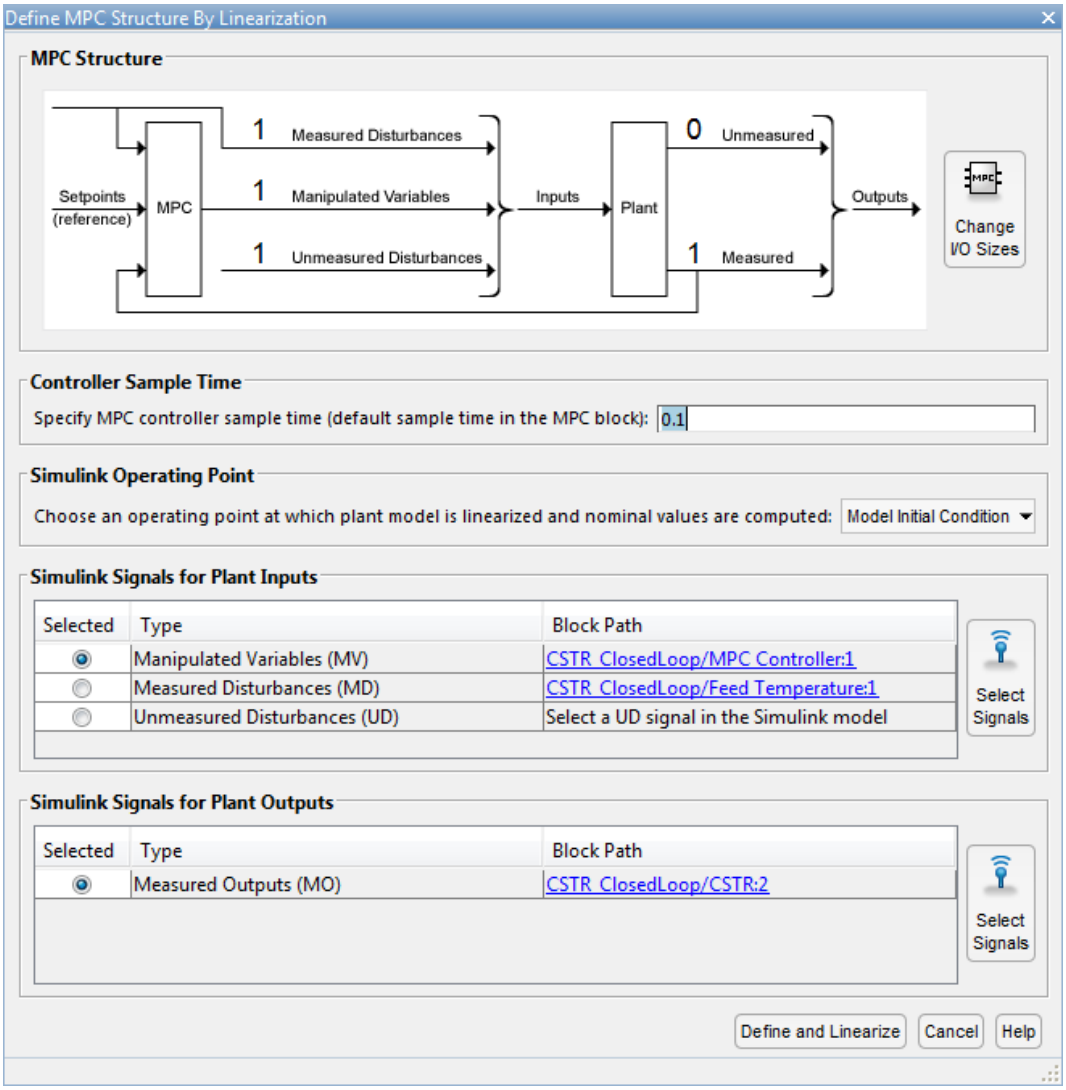

***Figure 5.7**: MPC model linearization in MPC Designer Toolbox®*

5.3.4 Horizons definition

With a model predictor identified and integrated with the controller structure, the next step is the tuning of the MPC. The first parameter to be tuned is the prediction horizon as is usually the recommended practice. Within the toolbox the interface of the tuning tab helps to properly define the value of the prediction horizon through trials and evaluations. As a standard scenario, the interface shows the unconstrained system output to a step response and visualizes the relative input: not only can the response of the system be studied but also the optimized input generated.

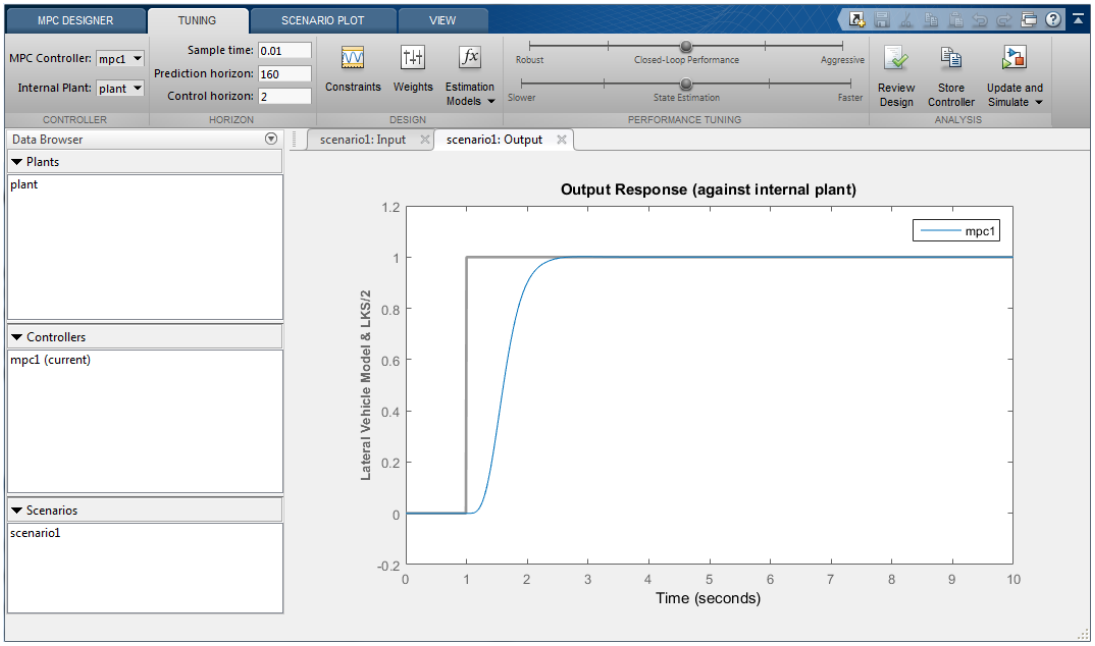

***Figure 5.8**: Tuning tab in MPC Designer Toolbox®*

The aim was to obtain a response with a fast transient but most importantly, given the application, a null overshoot: the vehicle is driven to the new reference trajectory (in this case a new reference value of q) in less than 2 seconds.

A control horizon of 2 units was sufficient to produce an optimal control action: as explained in previous sections this is a benefit of the receding horizon control whereby a larger value is only detrimental to the optimization task.

5.3.5 Constraints and weights optimization

Eventually the controller effectiveness and performance can be further improved by properly choosing constraints and weights.

The main constraint considered was the maximum torque exerted by the EPS actuator: for a passenger car this value ranges between 2 to 10 Nm depending on the car segment [48]. Given that these were just reference values, as more specific information was not available, a value of 7 was chosen. Other constraints available were the rate of steering torque, which again was left on default values for lack of references, and the manipulated output constraints.

While imposing limits on the vehicle center of mass position error (e.g. $|q| < 1$) may seem an intuitive choice in reality this overcomplicates the online optimization problem: the target is to obtain such good trajectory tracking performance without needing to impose any hard constraint on the solver itself.

Lastly, weights can be defined to better tune the system: in this case it was not necessary to change the default values except for the input rate weight for which a 0.1 value was beneficiary.

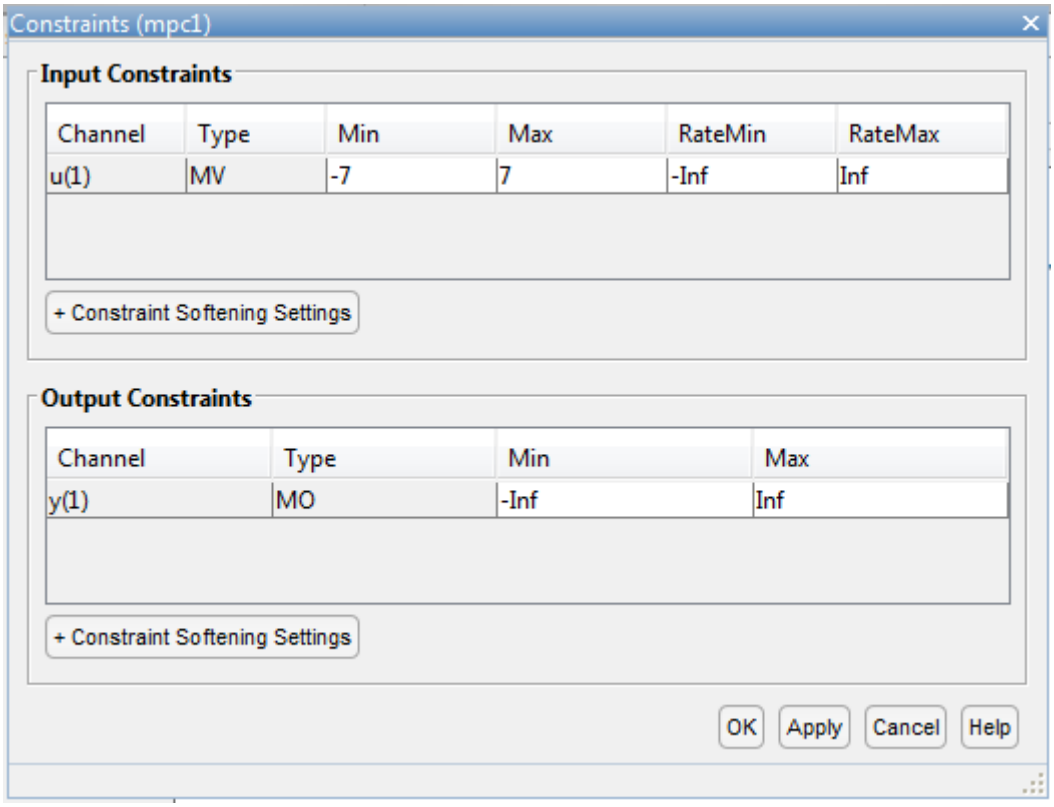

***Figure 5.9**: Constrains definition in MPC Designer Toolbox®*

6 Results and Discussion

Here we finally analyze the performance of the controllers designed, with particular focus on the lateral MPC controller being this the unique contribution of the present research work. The present predictive controller can be integral with a MPC based path planner: in this scenario the path planning algorithm would include the constraints and requirement of the control problem thus generating an already optimized path which would in turn benefit the MPC controller as well.

The controller designed within this work resulted to be stable for what concerns internal controller stability and closed-loop stability. The MPC Designer Toolbox allows to evaluate the stability of the system controller: in particular the eigenvalues of the discrete space-time realization of the closed-loop are evaluated to this scope.

6.1 Longitudinal Control

For this controller the target was simply that of tracking the vehicle speed exploiting a PI controller only. As it can be seen in the graphs optimizing the controller gains allowed to minimize the speed tracking error, which does not exceed 0.2 km/h for the whole duration of the simulation.

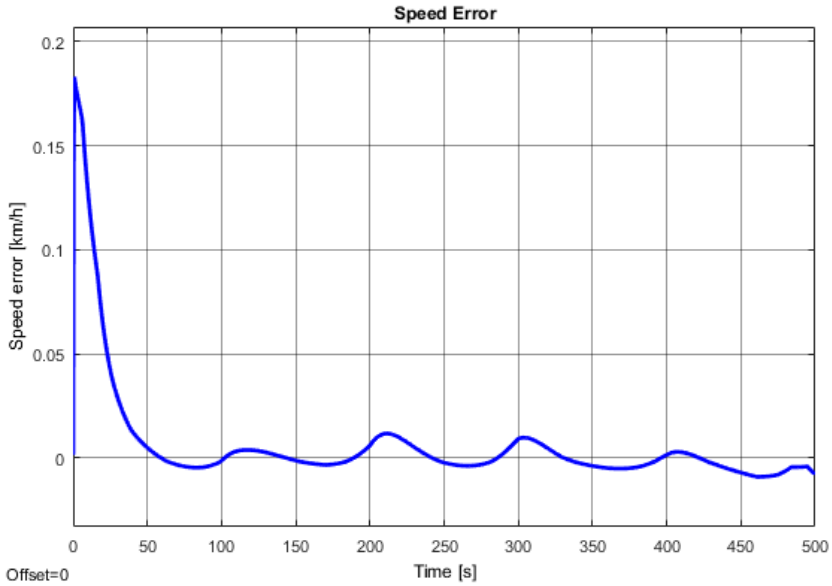

***Figure 6.1**: Speed Error resulting from the PI controller*

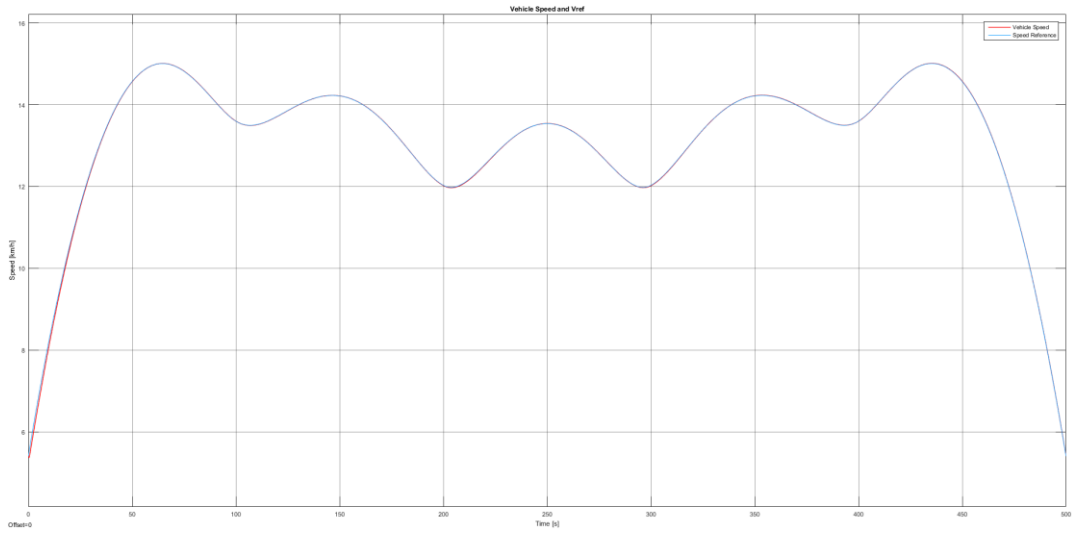

***Figure 6.2**: Comparison between reference speed and actual vehicle speed*

In the figure above the variating slow speed profiles of the reference and actual vehicle speeds are overlapping owing to the great tracking performance of the controller.

6.2 Lateral control

The stated objective of the present work was that of developing a full vehicle control strategy for AVP application: it is clear how the lateral control is what really matters in this application.

In order to create a demanding control problem moderate curvature trajectory was generated for the vehicle: this would allow to test the control strategy against realistic requirements.

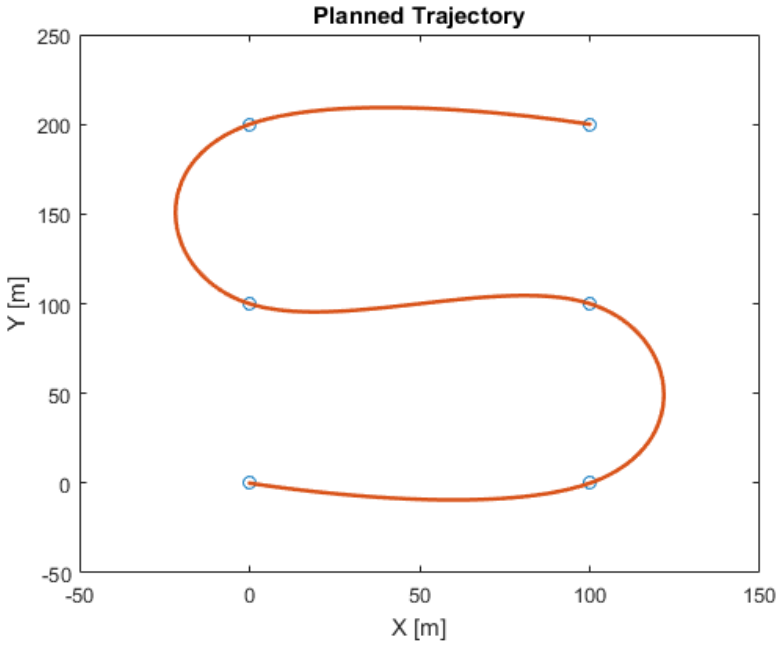

***Figure 6.3**: Trajectory employed for the lateral controller design*

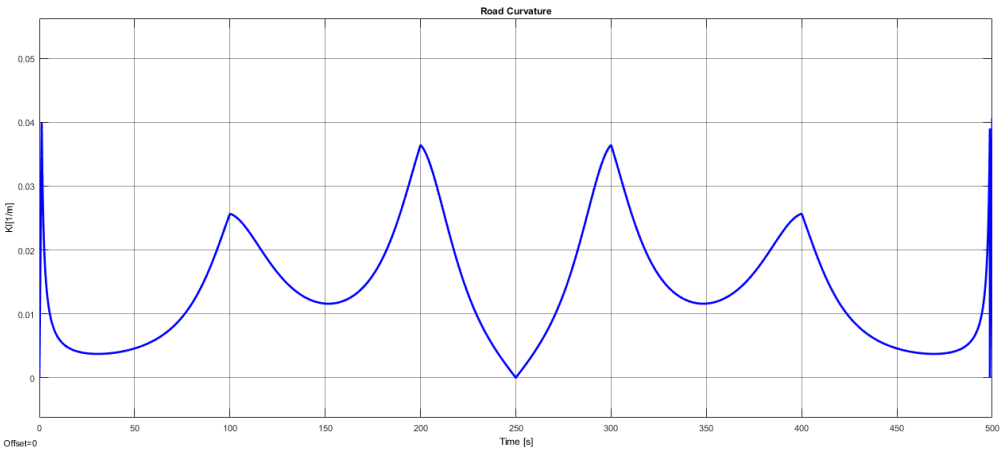

***Figure 6.4**: Curvature of the generated trajectory*

A smooth and simple trajectory allows to individuate limits of the controller when the generated control inputs are analyzed. We see how both the steering angle and torque produced by the controller are stable with smooth transitions even if some limited boundary oscillations. In particular the initial torque input appears to steer the vehicle in the opposite direction before transitioning quickly to the correct one.

This phenomenon, limited to small angles was seen in other simulation, and could due to the sensitivity of the EPS model.

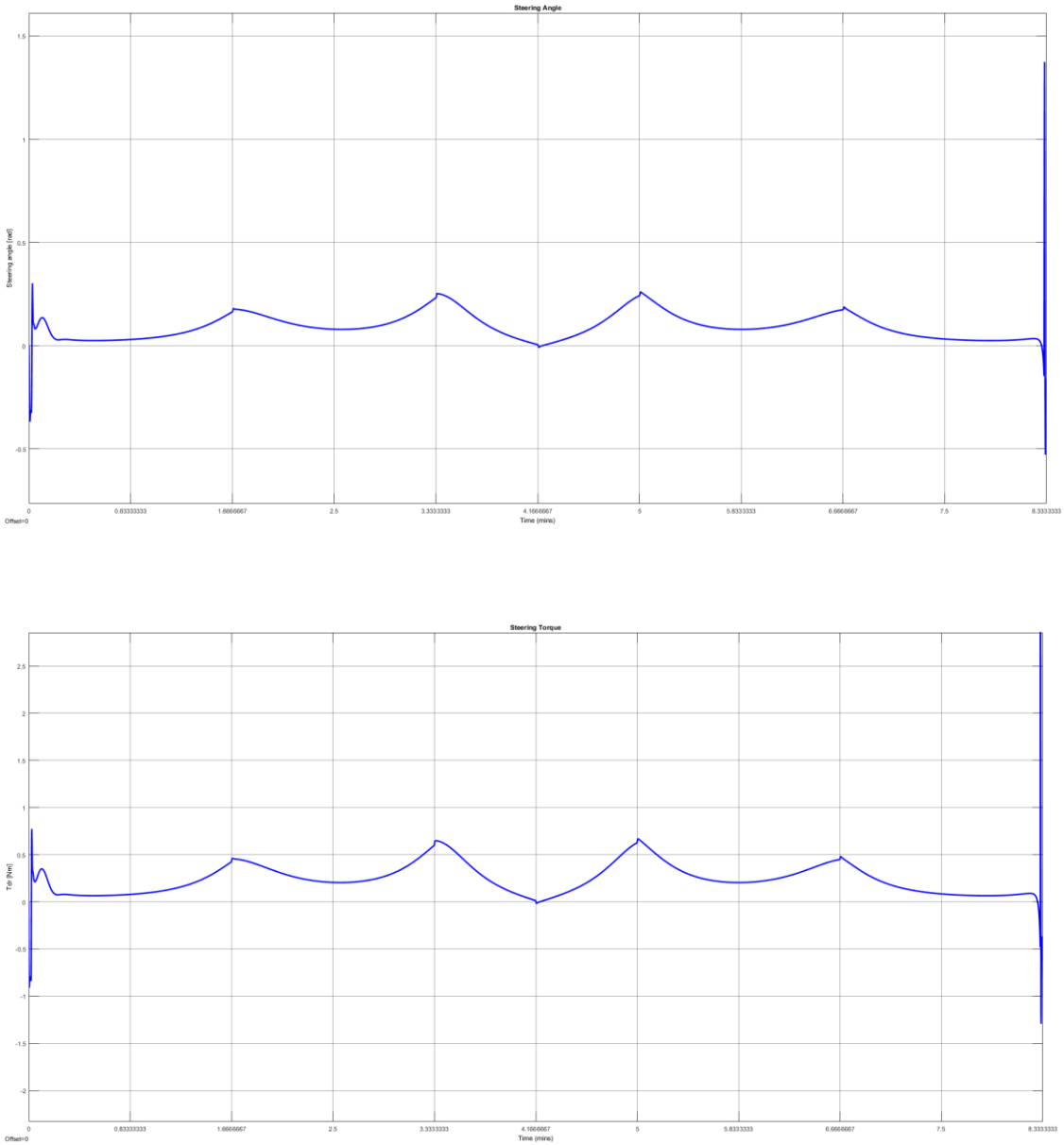

***Figure 6.5**: Steering Torque and Angle obtained from the MPC controller*

When analyzing the accuracy of lateral position tracking we found the relative position of the vehicle compared to the trajectory, or the local approximation of it, never exceeds 0.3m. Clearly in parking applications this may not be sufficient as more precise position control of the vehicle is needed due to presence of nearby vehicles and obstacles. Nevertheless, this is useful starting point for developing even more effective lateral control of the vehicle.

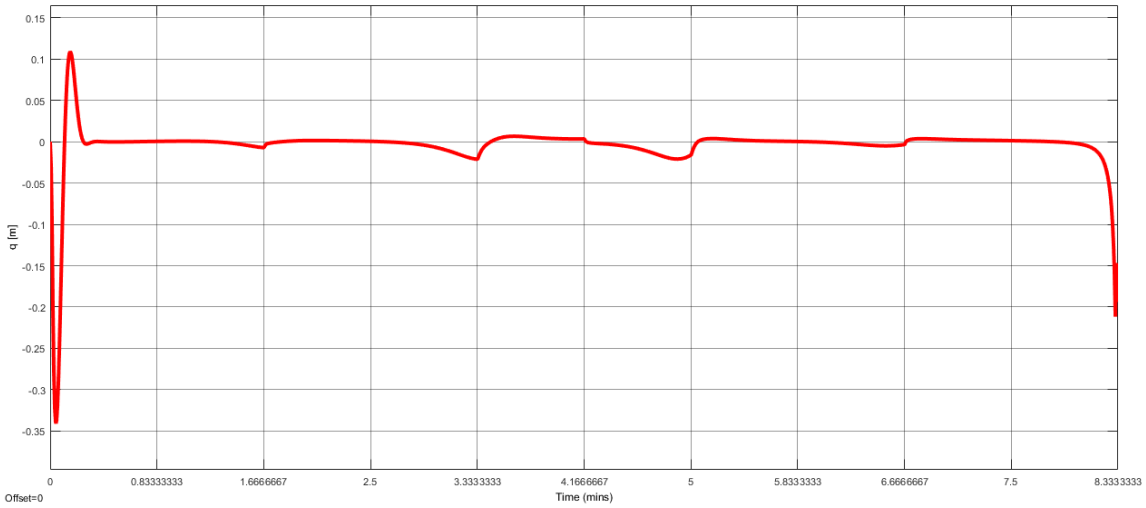

***Figure 6.6**: Vehicle center of gravity q as a function of time*

From the comparison of the driven trajectory with the reference trajectory the patter that emerges from q plot is confirmed: the initial uncertainty of the steering input deviated the vehicle from its path which is eventually regained and closed tracked for the remaining part. In the trajectory comparison we see how this uncertainty is actually limited to the very first meters of the driven path. In the last plot it is clear that the MPC based lateral control proves to be generally effective in driving the vehicle on the predefined path.

The instabilities present in the two extremities of the trajectory may also be a consequence of high curvature variations present in this part of the path: the curvature initially has a null value from the assumption of initial null slope and then it is quickly changing to a non-null value. Therefore, a proper re-definition of the curvature profile at the path extremities needs to be considered.

Eventually we need to evaluate the real-time performance of the controller: on average the simulation of 500s of length took a minute more than the simulated time to run. Clearly this does not satisfy real-time performance requirements.

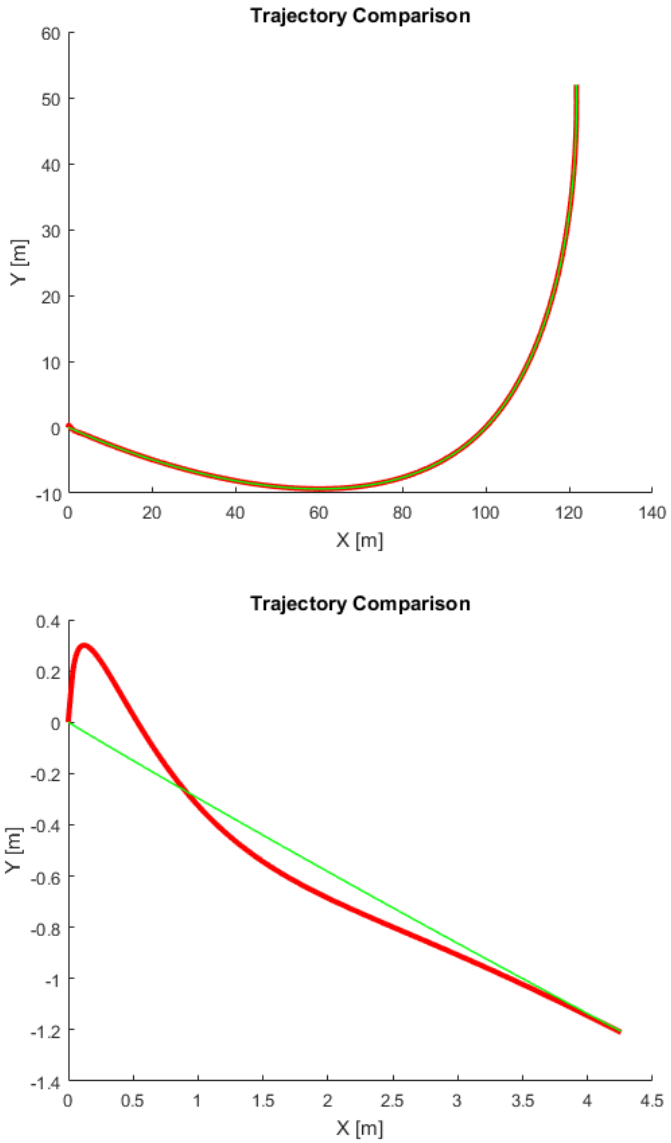

***Figure 6.7**: Comparisons between driven (red) and planned trajectories (green) with detailed view of initial trajectories.*

7 Conclusions and Future work

In this final chapter some conclusions are drawn and recommendations for future work are given.

From the final considerations in the last chapter and from some of the results it appears that using a truth model for this kind of application may be overcomplicating the problem: this emerges also from the need of using a sample time of 0.01s for the online optimization problem which greatly hinders real-time performance.

A longer sample time coupled with a simpler predictor model are then necessary to further improve the control performance.

For what concerns the accuracy of the system it seems that initial transients need to be deeply evaluated at each block to isolate the initial instability: as pointed out in the previous chapter a more accurate EPS model could be useful for this scope. One could obtain said model this from real data collected from vehicles running in parking scenarios: not only would this model be more accurate but also account for nonlinearities that could emerge at low speed and high steering angles.

Consolidated procedure in the industry would certainly include a detailed evaluation of the controller's robustness that was not performed here: before any validation in real scenario can be carried the virtual system need to prove robustness to slope and other disturbances. This is particularly important if upstream information flow in the system's pipeline does not deal effectively with external perturbation: when these reach the control problem they become even more laborious to manage.

From this consideration it appears that a successful control strategy can only be developed in close synergy with other modules in the AVP architecture: each stage's performance is closely affected by the other with an increasingly degrading effect on overall performance as inaccuracies are propagated downstream. This would indicate to fully integrate the path planning algorithm with the MPC controller to further enhance the performance of the whole system.

Finally, when validating the control system some comfort requirements that were not explicitly considered in this work could be included as well, such as limiting lateral accelerations and longitudinal jerk.

Bibliography

[1] https://www.nvidia.com/en-us/self-driving-cars/drive-platform/

[2] Lex Fridman (MIT), “6.S094: Deep Learning for Self-Driving Cars” January 2018, Lecture 2 slides

[3] https://en.wikipedia.org/wiki/Differential_GPS

[4] Rainer Kummerle, Dirk Hahnel, Dmitri Dolgov, Sebastian Thrun, Wolfram Burgard, “Autonomous Driving in a Multi-level Parking Structure” -

[5] SAE International Surface Vehicle Recommended Practice J3016 – ® Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles

[6] Thupakula, K., "Data Fusion Techniques for Object Identification in Airport Environment," SAE Technical Paper 2017-01- 2109, 2017, doi:10.4271/2017-01-2109.

[7] Jorge Sans Sangorrin, Jan Sparbert, Ulrike Ahlrichs and Wolfgang Branz-Robert Bosch GmbH Oliver Schwindt, “Sensor Data Fusion for Active Safety Systems” - 2010 SAE International

[8] Sebastian Klemm, Marc Essinger, Jan Oberlaender, Marc René Zofka, Florian Kuhnt, Michael Weber, Ralf Kohlhaas, Alexander Kohs, Arne Roennau, Thomas Schamm and J.Marius Zoellner , “Autonomous Multi-Story Navigation for Valet Parking” – 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)

[9] Jeong, Y., Kim, S., Yi, K., Lee, S. et al., "Design and Implementation of Parking Control Algorithm for Autonomous Valet Parking," SAE Technical Paper 2016-01-0146, 2016, doi:10.4271/2016-01-0146.

[10] Bernd Schaeufele, Oliver Sawade, Fraunhofer Institute for Open Communication Systems (FOKUS) and Daniel Becker, Ilja Radusch Daimler Center for Automotive Information Technology Innovations (DCAITI) Berlin, Germany,” A transmission protocol for fully automated valet parking using DSRC” - 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC)

[11] Seung-Jun Han and Jeongdan Choi, “Parking Space Recognition for Autonomous Valet Parking Using Height and Salient Line Probability Maps”

[12] Christian Unger, Eric Wahl, Slobodan Illic, “Parking assistance using dense motion-stereo” - Machine Vision and Applications (2014) 25:561–581 DOI 10.1007/s00138-011-0385-1

[13] Peter Muehfellner, Paul Furgale, Wojciech Derendarz and Roland Philippsen, “Evaluation of Fisheye-Camera Based Visual Multi-Session Localization in a Real-World Scenario – 2013 IEEE Intelligent Vehicles Symposium (IV)

[14] P. Furgale et al, “Toward Automated Driving in Cities using Close-to-Market Sensors – an overview of the V-Charge project” – 2013 IEEE Intelligent Vehicles Symposium (IV)

[15] Lim, W., Kim, J., Jo, K., Jo, Y. et al., "Turning Standard Line (TSL) Based Path Planning Algorithm for Narrow Parking Lots," SAE Technical Paper 2015-01-0298, 2015, doi:10.4271/2015-01-0298.

[16] Wu, L., Zha, H., Xiu, C., and He, Q., "Local Path Planning for Intelligent Vehicle Obstacle Avoidance Based on Dubins Curve and Tentacle Algorithm," SAE Technical Paper 2017-01-1951, 2017, doi:10.4271/2017-01-1951.

[17] Xiong, S., Tan, G., Guo, X., Yang, M. et al., "Driving Path Planning System under Vehicular Active Safety Constraint," SAE Technical Paper 2016-01-8105, 2016, doi:10.4271/2016-01-8105.

[18] Changliu Liu, Yizhou Wang and Masayoshi Tomizuka, “Boundary Layer Heuristic for Search-based Nonholonomic Path Planning in Maze-Like Environments” – 2017 IEEE Intelligent Vehicles Symposium (IV)

[19] Kyoungwook Min and Jeongdan Choi, “A Control System for Autonomous Vehicle Valet Parking” - 2013 13th International Conference on Control, Automation and Systems (ICCAS 2013)

[20] Prasanth Jeevan, Frank Harchut, Bernhard Mueller-Bessler, and Burkhard Huhnke, “Realizing Autonomous Valet Parking with Automotive Grade Sensors” – The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems

[21] Holger Banzhaf, Dennis Nienh, Steffen Knoop, J. Marius “The Future of Parking: A Survey on Automated Valet Parking with an Outlook on High Density Parking” - Zollner 2017 IEEE Intelligent Vehicles Symposium

[22] Kyungbok Sung, Jungdan Choi, Dongyong Kwak, “Vehicle Control System for Automatic Valet Parking with Infrastructure Sensors” - 2011 IEEE International Conference on Consumer Electronics (ICCE)

[23] Sebastian Klaudt, Adrian Zlochi and Lutz Eckstein, “A-Priori Map Information and Path Planning for Automated Valet Parking – 2017 IEEE Intelligent Vehicles Symposium

[24] Jong Min Kim, Kyung Il Lim and Jung Ha Kim, “Auto Parking Path Planning System Using Modified Reeds-Shepp Curve Algorithm” –The 11th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI 2014)

[25] Christian Löper, Claas Brunken, George Thomaidis, Stephan Lapoehn, Paulin Pekezou Fouopi, Henning Mosebach and Frank Köster, “Automated Valet Parking as Part of an Integrated Travel Assistance” – 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013),

[26] B.Song, “Cooperative Lateral Vehicle Control for Autonomous Valet Parking” – International Journal of Automotive Technology, Vol. 14, No. 4, pp. 633−640 (2013) DOI 10.1007/s12239−013−0068−1

[27] Yang, W., Zheng, L., Li, Y., Ren, Y. et al., "A Trajectory Planning and Fuzzy Control for Autonomous Intelligent Parking System," SAE Technical Paper 2017-01-0032, 2017, DOI 10.4271/2017-01-0032.

[28] Yokoyama, K., Iezawa, M., Akashi, Y., Satake, T. et al., "Speed Control of Parking Assist System for Electrified Vehicle," SAE Technical Paper 2015-01-0316, 2015, DOI 10.4271/2015-01-0316.

[29] Hafner, M. and Bales, J., "Low-Speed Longitudinal Distance Control with Applications to Parking," SAE Technical Paper 2017-01-0036, 2017, doi:10.4271/2017-01-0036.

[30] Jeong, Y., Kim, S., Yi, K., Lee, S. et al., "Design and Implementation of Parking Control Algorithm for Autonomous Valet Parking," SAE Technical Paper 2016-01-0146, 2016, doi:10.4271/2016-01-0146.

[31] Seung-Jun Han, Juwan Kim, and Jeongdan Choi, “Effective Height-Grid Map Building using Inverse Perspective Image” 2015 IEEE Intelligent Vehicles Symposium (IV) June 28 - July 1, 2015. COEX, Seoul, Korea

[32] Bai Li, KexinWang, and Zhijiang Shao, “Time-Optimal Maneuver Planning in Automatic Parallel Parking Using a Simultaneous Dynamic Optimization Approach”, IEEE transactions on intelligent transportation systems, vol. 17, no. 11, november 2016

[33] Mihai Chirca and Guillaume Martin (Renault – ADAS), Roland Chapuis, Christophe Debain, Ronald Lenain, “Autonomous Valet Parking System Architecture”, 2015 IEEE 18th International Conference on Intelligent Transportation Systems

[34] Bongsob Song, Dohyun Kim, and Heejae Choi, “Cooperative Lateral Control for Automatic Valet Parking” - 2011 11th International Conference on Control, Automation and Systems

[35] Stephan Rottmann, Julian Timpner, and Lars Wolf, “Demo: Automated Valet Parking and Charging”, 2014 IEEE Vehicular Networking Conference (VNC)

[36] Kyoung-Wook Min and Jeong-Dan Choi,”Design and Implementation of an Intelligent Vehicle System for Autonomous Valet Parking Service”,2015 IEEE

[37] Holger Banzhaf, Frank Quedenfeld, Dennis Nienhueser, Steffen Knopp, J.Marius Zoellner,” High Density Valet Parking Using k-Deques in Driveways” - 2017 IEEE Intelligent Vehicles Symposium (IV)

[38] Marco Allodi, Luca Castangia, Alessandro Cionini, Francesco Valenti, “Monocular Parking Slots and obstacles detection and tracking” - 2016 IEEE Intelligent Vehicles Symposium (IV) Gothenburg, Sweden, June 19-22, 2016

[39] David C. Conner, Hadas Kress-Gazit, Howie Choset,Alfred A. Rizzi, and George J. Pappas, “Valet Parking Without a Valet” - 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems

[40] Dong Hee Kang, Chang Mook Kang, Joong-Sik Kim, Seunghyun Kim, Whoi-Yul Kim, Seung-Hi Lee, and Chung Choo Chung, “Vision-based Autonomous Indoor Valet Parking System” - 2017 17th International Conference on Control, Automation and Systems (ICCAS 2017)

[41] Xu Y. Chen B.Y. Shan X. Jia W.H. Lu Z.F. Xu G., “Model Predictive Control for Lane Keeping System in Autonomous Vehicle” College of Mechatronics and Control Engineering, Shenzhen University, China

[42] Marcus Nolte, Marcel Rose, Torben Stolte and Markus Maurer, “Model Predictive Control Based Trajectory Generation for Autonomous Vehicles – An Architectural Approach”- 2017 IEEE Intelligent Vehicles Symposium (IV) June 11-14, 2017, Redondo Beach, CA, USA

[43] Chao Huang, Fazel Naghdy, Haiping Du, “Model Predictive Control-based Lane Change Control System for An Autonomous Vehicle”- 2016 IEEE Region 10 Conference (TENCON)

[44] T. D. Chu and C. K. Chen, “Modelling and model predictive control for a bicycle-rider System”- VEHICLE SYSTEM DYNAMICS, 2018 VOL. 56, NO. 1, 128–149

[45] Rajesh Rajamani - Vehicle Dynamics and Control (2012, Springer US) Mechanical Engineering Series

[46] V. Cerone, M. Milanese and D. Regruto, “Combined Automatic Lane-Keeping and Driver's Steering Through a 2-DOF Control Strategy” - IEEE Transactions on Control Systems Technology, Vol. 17, No. 1, January 2009.

[47] M. F. Land and D. Lee. – “Where we look when we steer”, Nature 1994

[48] https://www.bosch-mobility-solutions.com/media/global/products-and-services/passenger-cars-and-light-commercial-vehicles/steering-systems/servolectric-steering-systems/folder_for_steering_systems_pkw.pdf

[49] MathWorks® Video and Webinar Series “Understanding Model Predictive Control” at: https://www.mathworks.com/videos/series/understanding-model-predictive-control.html

Vita Auctoris

NAME:	Oussama Erraji
PLACE OF BIRTH:	Rabat, MOROCCO
YEAR OF BIRTH:	1993
EDUCATION:	Politecnico di Torino, B.Sc. in Automotive Engineering, Torino, Italy, 2016
	Politecnico di Torino, M.Sc. in Automotive Engineering, Torino, Italy, 2018
	University of Windsor, International M.A.Sc. in Automotive Engineering Windsor, Canada, 2018