university of groningen

faculty of science and engineering

MASTER'S RESEARCH THESIS

Spatial-Based Model Predictive Control for Trajectory Generation in Autonomous Racing

Author:
S. DE BOER
S2598116

1st Supervisor:
prof. dr. B. JAYAWARDHANA
2nd Supervisor:
dr. N. MONSHIZADEH NAINI

A research thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

in

Industrial Engineering and Management
University of Groningen

March 25, 2020

Abstract

Faculty of Science and Engineering

University of Groningen

Master of Science

Spatial-Based Model Predictive Control for Trajectory Generation in Autonomous Racing

by S. DE BOER

An attractive research direction within automation is autonomous vehicles and especially in the last decade, applications in this topic have undergone tremendous improvement. Within the field of vehicle control, autonomous racing has gained increased interest in terms of research. Understanding how a race car driver controls a vehicle at its friction limits can provide insights into the development of vehicle safety systems.

An integrated plan-and track control algorithm to automatically steer an autonomous race car along an optimized desired trajectory is designed, using Model Predictive Control. A kinematic bicycle model is employed to derive the dynamics of the vehicle. Commonly used time-dynamics are reformulated to spatial dynamics by expressing equations in terms of the center-line of the track and are used to design the controller. Hence, the name Spatial-Based Model Predictive Control (MPC). The utmost advantage of this reformulation is the allowance of natural modeling of static and dynamic objects (representing race opponents) as spatial bounds on the track.

The nonlinear vehicle dynamics are successively linearized around previous solutions of the MPC problem, allowing the problem to be formulated as a convex optimization problem. The objective of the Spatial-Based MPC controller is to maximize the progress along the center-line of a given track, while ensuring smoothness along the way. The performance of the controller is tested on manually created tracks representing several segments of the F1 Suzuka Circuit in Japan. Several simulations are executed to validate and show robustness of the Spatial-Based MPC controller, also in the presence of race opponents.

Contents

Abstract **iii**

1 Introduction **1**
1.1 Research Motivation 2
1.1.1 Autonomous Vehicle System 2
1.1.2 Path Planning and Trajectory Tracking 2
1.2 Research Scope 4
1.3 Research Goal 5
1.4 Thesis Outline 5

2 Literature Study **7**
2.1 Autonomous Vehicle System 7
2.1.1 Sensors 8
2.1.2 Vehicle Overtaking Manoeuvre 8
2.2 Vehicle Model Strategies 9
2.2.1 Geometric 9
2.2.2 Kinematic 10
2.2.3 Dynamic 12
2.3 Path Planning & Trajectory Control at the limit of friction 14
2.3.1 Racing line 15

3 Vehicle Model **17**
3.1 Kinematic Bicycle Model 17
3.2 Spatial Dynamics Reformulation 19

4 Controller Design **23**
4.1 Model Predictive Control (MPC) 23
4.2 Control Ojective 24
4.2.1 Racing Behaviour 24
4.2.2 Smoothness 26
4.2.3 Constraints 26
4.3 Linear Space Varying (LSV)-Model 27
4.3.1 Linearization 27
4.3.2 Discretization 28
4.4 Final Control Structure 29

5 Simulation Setup **33**
5.1 Nonholonomic vs. holonomic robots 33
5.2 Simulation Environment 33

5.3 F1 Suzuka Circuit Japan . . . 34

6 Simulation and Results **35**

6.1 Validation . . . 35

6.1.1 Validation of J_{race} . . . 35

6.1.2 Validation of J_{smooth} . . . 36

6.2 Friction Coefficient μ . . . 38

6.2.1 μ on S-curve . . . 38

6.2.2 μ on Hairpin Curve . . . 39

6.3 Robustness Analysis . . . 39

6.3.1 Prediction Horizon . . . 39

6.3.2 Track Variation . . . 41

6.3.3 Initial Conditions . . . 41

6.3.4 Overtaking Manoeuvres . . . 42

7 Discussion **47**

8 Conclusion **49**

9 Future Research **51**

A Convexity Definitions (Princeton University, 2020) **53**

B MPC Matrix Derivation **55**

B.1 Taylor Series . . . 55

B.2 Linearization . . . 56

B.3 Discretization . . . 57

C Additional track information **59**

C.1 Track 1: S-curve . . . 59

C.2 Track 2: 90-degrees curve . . . 60

C.3 Track 3: Hairpin curve . . . 60

D Trajectories for changing friction coefficient μ **61**

D.1 change in μ on S-curve . . . 61

D.2 Change in μ on hairpin curve . . . 63

E Varying Horizon Lengths N and Step Lengths Δs **65**

E.1 Constant $N = 5$ and varying Δ_s . . . 65

E.2 Constant $\Delta_s = 4$ and varying N . . . 66

Bibliography **67**

List of Figures

1.1 SAE classification for autonomous vehicles . . . 3
1.2 Core competencies of a typical autonomous vehicle system with system boundaries . . . 4

2.1 Overview of autonomous vehicle system (Amer et al., 2017) . . . 8
2.2 Overview of a basic overtaking manoeuvre . . . 9
2.3 Vehicle Axis System used in Vehicle Modelling (Kissai et al., 2019) . . . 10
2.4 Ackermann steering configuration (Amer et al., 2017) . . . 10
2.5 Kinematic vehicle models . . . 11
2.6 Dynamic vehicle models . . . 12
2.7 General control architecture for an autonomous vehicle . . . 14
2.8 Effect of using Centre of Percussion on a dynamic bicycle model . . . 15
2.9 Perfect racing line . . . 16

3.1 Considered kinematic bicycle model in global- and curvilinear coordinate system . . . 19
3.2 Visualization of inertial coordinates . . . 20

5.1 F1 Suzuka Circuit, Japan . . . 34
5.2 Representation of Track Segments . . . 34

6.1 Validation of objective function J_{race} . . . 36
6.2 Validation of J_{smooth} . . . 37
6.3 MPC iterations vs. spatial states . . . 37
6.4 Variation of friction values μ on S-curve . . . 38
6.5 Variation of friction values μ on Hairpin curve . . . 39
6.6 Constant $N = 5$ and changing Δ_s . . . 40
6.7 Constant $\Delta_s = 4$ and changing N . . . 41
6.8 Simulation on different track segments . . . 41
6.9 Variation in initial values . . . 42
6.10 Scenario 1: one race opponent . . . 43
6.11 Scenario 2: two race opponents . . . 44
6.12 Scenario 3: three race opponents . . . 45
6.13 Computation times in the presence of race opponents . . . 45

A.1 Convexity of sets . . . 53
A.2 Illustration of definition of convexity of function . . . 54

C.1 Track information S-curve . . . 59
C.2 Track information 90-degrees curve . . . 60

C.3 Track information hairpin curve 60

D.1 From left to right: $\mu = 0, \mu = 0.5, \mu = 0.8, \mu = 1$ 61
D.2 From left to right: $\mu = 2, \mu = 3, \mu = 10$ 62
D.3 From left to right: $\mu = 0, \mu = 0.5, \mu = 0.8, \mu = 1, \mu = 2, \mu = 3, \mu = 10$. 63

E.1 S-curve trajectory with constant $N = 5$ and changing Δ_s 65
E.2 S-curve trajectory with constant $N = 5$ and changing Δ_s 66

List of Tables

1.1 Levels of vehicle autonomy classified by the SAE 2

3.1 Spatial States of the vehicle model 22
3.2 Control inputs of the spatial vehicle model 22

6.1 Initial values for validation 36
6.2 Effect of varying planning horizon and step length 40

List of Abbreviations

ACC	**A**daptive **C**ruise **C**ontrol
ADAS	**A**dvanced **D**river **A**ssistence **S**ystems
AGV	**A**utonomous **G**round **V**ehicle
CG	**C**enter of **G**ravity
LSV	**L**inear **S**pace **V**arying
LTV	**L**inear **T**ime **V**arying
MPC	**M**odel **P**redictive **C**ontrol
(R)DW	**D**ienst **W**egverkeer
RWD	**R**ear **W**heel **D**riven
SAE	**S**ociety (of) **A**utomotive **E**ngineers
UAV	**U**nmanned **A**erial **V**ehicle
UUV	**U**nmanned **U**nderwater **V**ehicle
V2V	**V**ehicle (to) **V**ehicle
ZOH	**Z**ero **O**rder **H**old

List of Symbols

Δ_s	step length	m
N	prediction horizon	-
μ	friction coefficient	-
δ	steering angle	rad
κ_v	car curvature	$\mathrm{m^{-1}}$
ψ	yaw angle	rad
$\dot{\psi}$	yaw rate	$\mathrm{rad\ s^{-1}}$
v_x	longitudinal velocity	$\mathrm{m\ s^{-1}}$
C_v	curvature rate	$\mathrm{rad\ m^{-2}}$
E_y	lateral deviation w.r.t. center-line track	m
E_y	angular deviation w.r.t. center-line track	rad
a_x	longitudinal acceleration	$\mathrm{m\ s^{-2}}$
s	spatial coordinate	m
$\dot{s}$	velocity along center-line track	$\mathrm{m\ s^{-1}}$
ζ	state vector	-
u	control vector	-
κ_s	road curvature	$\mathrm{m^{-1}}$

Chapter 1

Introduction

The use of autonomous systems have become prevalent and are used to perform tasks that are originally considered mundane, dangerous or time-consuming. Increased programming technology entails an increased demand of robotic devices able to work in complex environments without human assistance. On the one hand this trend has been unfavourable for a huge amount of people, as millions of jobs has been lost to automation already (Forbes, 2019). On the other hand, innovative activities in the automatic control field has led to a significant boost in research interest. An attractive research direction within automation is autonomous vehicles and especially in the last decade, this topic has undergone tremendous improvement. As knowledge of autonomous vehicles and its applications proliferates, so does the deployment of these vehicles. For example, governmental institutions invest heavily in the development of these technologies for military purposes. The application of Unmanned Aerial Vehicles (UAV) for area surveillance in the air or Unmanned Underwater Vehicles (UUV) for underwater mine detection, are common actions taken by respectively military forces and the navies (Angelov, 2012).

This research focuses on vehicles for terrestrial purposes only, which are commonly referred to as Autonomous Ground Vehicles (AGV). For this particular field, a large number of research directions have been taken in the past. Interesting concepts like car platooning or convoying (Amoozadeh et al., 2015), lane changing, emergency stopping, obstacle avoidance and overtaking manoeuvres (Ozguner, Acarman, and Redmill, 2011) have been defined over the last decade. Two main aspects contributing to these topics are path planning and trajectory tracking, for which many control techniques are proposed in literature. State-of-the art autonomous vehicles have demonstrated a great deal of successes, but still cannot compete with human drivers in many ways, leading to interesting challenges to solve. One particular challenge for autonomous vehicles, induced by professional race car driving, is vehicle control at the limit of friction. Race car drivers have the ability to operate their car at top speed, while the tires experience frictions at their limits, without losing control by utilising their internal vehicle model to coordinate steering, throttle and brake inputs (Kritayakirana and Gerdes, 2012).

1.1 Research Motivation

1.1.1 Autonomous Vehicle System

Advanced Driver-Assistance Systems (ADASs) were designed to reduce or even eliminate traffic errors and for the enhancement of overall traffic and transport efficiency (Brookhuis, De Waard, and Janssen, 2019). The continuing evolution of technology in the automotive industry aims to deliver systems like these with even greater safety benefits, such that that one day fully automated vehicles drive us in stead of us driving them. A vision that, considering the proliferation within this branch of technology, seems to be on the verge of becoming reality (NHTSA, 2018). Research in the field of ADASs has been conducted for several decades and has shown that, by minimizing human errors, less road fatalities occurs when an ADAS is employed (Cafiso and Di Graziano, 2012), (Pascual and Pablo, 2009). Since autonomous safety features are frequently used and referred to nowadays, the US Society of Automotive Engineers (SAE) provided an infographic chart to classify autonomous vehicle systems into six different levels, as can be seen in Figure 1.1 (SAE, 2018). Additionally, Table 1.1 summarizes the classification from no automation to full automation performed by the Automated Driving System (ADS) (NHTSA, 2018).

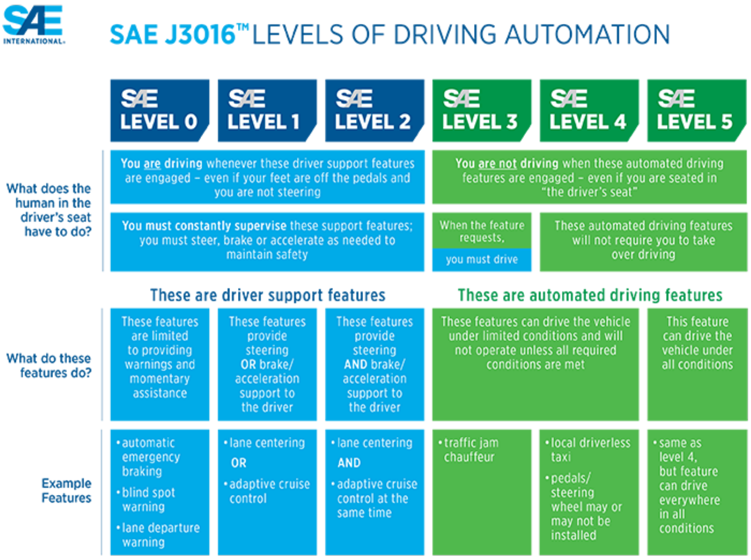
	Classification	Description
Level 0	No Automation	Driver performs all driving tasks
Level 1	Driver Assistance	Driver has hands on the wheel
Level 2	Partial Automation	Hands off the wheel, eyes on the road
Level 3	Conditional Automation	Hands off the wheel, eyes off the road
Level 4	High Automation	ADS performs all driving tasks
Level 5	Full Automation	Humans are passengers

TABLE 1.1: Levels of vehicle autonomy classified by the SAE

This research is done in collaboration with Dienst Wegverkeer (RDW), which is an organization that approves and licenses vehicles and vehicle parts in The Netherlands (RDW, 2018). Since the number of vehicles with autonomous features expands rapidly, RDW is curious about the working principles of control software implemented in autonomous cars. Recently, RDW launched a self-driving challenge in which a go-kart was targeted to drive autonomous by means of line detection. This technique is rather simplistic in this form, as the car follows solely a reference line without any other moving obstacles present. Since this situation is not in line with real-life occurrences, a more in-depth research is done in the context of this thesis. Topics such as location, automatic map construction, monitoring, path planning and trajectory tracking for a certain generated environment has increased research interest due to their overlap with real-world situations. Especially trajectory planning and tracking constitute two important operations of an autonomous vehicle system (Ozguner, Acarman, and Redmill, 2011) and is elaborated on further in the next section.

1.1.2 Path Planning and Trajectory Tracking

As can be seen in Figure 1.2, the core competencies of an autonomous vehicle software system can be roughly categorized into three components: *perception*, *planning* and *control*. The systems perception refers to the process of converting incoming raw



SAE J3016™ LEVELS OF DRIVING AUTOMATION

	SAE LEVEL 0	SAE LEVEL 1	SAE LEVEL 2	SAE LEVEL 3	SAE LEVEL 4	SAE LEVEL 5
What does the human in the driver's seat have to do?	**You are driving** whenever these driver support features are engaged – even if your feet are off the pedals and you are not steering			**You are not driving** when these automated driving features are engaged – even if you are seated in "the driver's seat"		
	You must constantly supervise these support features; you must steer, brake or accelerate as needed to maintain safety			When the feature requests, you must drive	These automated driving features will not require you to take over driving	
	These are driver support features			These are automated driving features		
What do these features do?	These features are limited to providing warnings and momentary assistance	These features provide steering **OR** brake/ acceleration support to the driver	These features provide steering **AND** brake/ acceleration support to the driver	These features can drive the vehicle under limited conditions and will not operate unless all required conditions are met		This feature can drive the vehicle under all conditions
Example Features	• automatic emergency braking • blind spot warning • lane departure warning	• lane centering **OR** • adaptive cruise control	• lane centering **AND** • adaptive cruise control at the same time	• traffic jam chauffeur	• local driverless taxi • pedals/ steering wheel may or may not be installed	• same as level 4, but feature can drive everywhere in all conditions

FIGURE 1.1: SAE classification for autonomous vehicles

data from the sensors or Vehicle to Vehicle (V2V) communication to recognition of certain objects (e.g. another car or a pedestrian) in the surrounding environment, i.e. a representation of the environment. It is analogous to how the human brain processes information through sight and it counts as input for the planning process. The planning system is the ability of the autonomous vehicle to make certain decisions to achieve predefined goals. Doing so, the vehicle knows how to act in a certain situation such as slowing down, turning left/right etc. Lastly, the control system is related to the process of converting goals and intentions from the planning system into actual actions. Currently, high-end vehicles are equipped with autonomous safety systems. Features such as assisted braking, line detection and following enhance the safety of these high-end vehicles. However, these applications can solely help in trivial driving situations. In reality, driving situations are highly determined by human inputs and their driving habits (Schwarting et al., 2017). In terms of software language, these driving habits are controlled by path planning and trajectory tracking and these terms will be briefly discussed in this section.

Path planning can be defined as the computation of desired actions of motion profiles for autonomous machines (Biagiotti and Melchiorri, 2008). Autonomous vehicles are dependent on the real-time state of the vehicle and information about its surrounding such that a local path is derived that ensures a safe drive while deviation from the global trajectory (the overall journey) is minimized. According to (Katrakazas et al., 2015), local path planning is defined as real-time planning of the vehicle's transition from one feasible state to the next, while satisfying the vehicle's kinematic limits based on vehicle dynamics and constrained by occupant comfort, lane boundaries and traffic rules, while, at the same time, avoiding obstacles. Different strategies for planning an path for overtaking can be found in literature. Classical techniques such as potential fields, cell-decomposition, interdisciplinary methods and optimal control have the drawback of not being able to include the vehicle dynamic. A relatively new concept for local path planning is the employment of Model Predictive Control (MPC). Many researchers favour this method for path planning, due to its ability to more efficient handling of system constraints, vehicle dynamics and non-linearities. The working principles of this approach will be further elaborated on in Chapter 3.

Trajectory tracking is referred to as computing the actuator input that enables the autonomous vehicle to track a desired trajectory. In trajectory tracking control, the longitudinal- and lateral movements of the vehicle is governed along a specified path or trajectory. It is a popular and mature scientific field with a profusion of control methods available in literature. Assessment of tracking controllers for autonomous vehicles is executed through properties such as real-time capability, robustness, operating range and control parameter tuning (Dixit et al., 2018). Same as for path planning control, MPC can be utilised in trajectory tracking controllers and has been found to perform quite well and provide accurate tracking performance. Due to its ability to include vehicle- and tire dynamics, MPC is often used in control problems for complex and high-speed driving situations. Handling a vehicle at its limit of friction is an example of such a complex situation. Extensive research is done on enabling the control of autonomous vehicles when tires experience friction at their limits, but still much knowledge can be gained from understanding how race car drivers maintain both a tracked path and car stability, especially when operating at the limit of friction.

1.2 Research Scope

As explained previously, this research scopes down on the path planning and trajectory control of an autonomous vehicle system. Within a typical autonomous vehicle system, the boundaries can be indicated as in Figure 1.2. Within the field of vehicle control, autonomous racing has gained increased interest in terms of research. Understanding how a race car driver controls a vehicle at its friction limits can provide insights into the development of vehicle safety systems during manoeuvres that require full handling capabilities.

As stated, path planning and trajectory tracking of autonomous vehicles have numerous control strategies. In this research, MPC will be applied to effectively generate trajectories in race situations. Several track segments will be manually created and simulated on with Python, to find the optimal track by optimizing a defined objective with the MPC method. By doing so, an integrated path planning and trajectory tracking control problem for autonomous racing is desired to be solved.

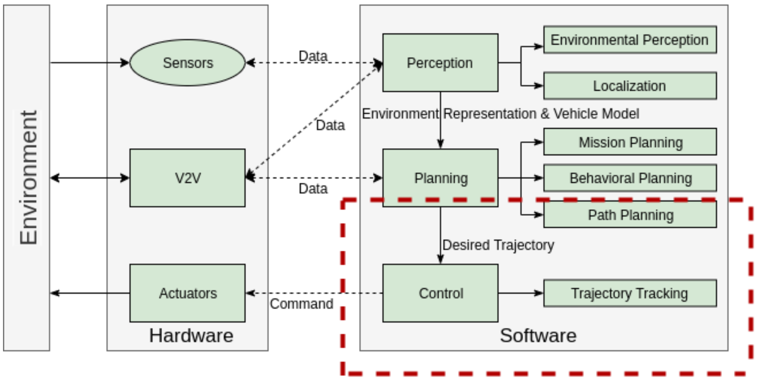


FIGURE 1.2: Core competencies of a typical autonomous vehicle system with system boundaries

1.3 Research Goal

In line with the previous sections, the research goal is to *design a plan-and track control algorithm to automatically steer an autonomous race car along an optimized desired trajectory, using model predictive control*.

A main research question and, inherently, some underlying sub-questions will provide the necessary guidance in achieving the above-mentioned goal:

How to generate optimal trajectories for autonomous overtaking using Model Predictive Control?

- *SQ1:* Which vehicle model is suitable for trajectory generation in autonomous racing?
- *SQ2:* How is the trajectory generation control algorithm designed?
- *SQ3:* How robust is the proposed control algorithm?

1.4 Thesis Outline

The remainder of this thesis is organized as follows. Chapter 2 gives an overview of the researched literature for convenient background information. The proposed vehicle model, fundamental in Model Predictive Control, is given in chapter 3. Subsequently, the implementation of the vehicle dynamics within the MPC scheme is elaborated on in chapter 4. The simulation setup is given in chapter 5 en the consequent simulations and results in chapter 6. Chapter 7 is utilized to discuss these results, whereupon final conclusions and future work are given in respectively chapter 8 and 9.

Chapter 2

Literature Study

This chapter provides an overview of the studied topics useful throughout this thesis. Section 2.1 elaborates in more detail on the complexity and components of autonomous driving. Section 2.2 gives an overview of different strategies to model vehicle dynamics. From these strategies, the best approach for the problem at hand will be chosen and justified. Subsequently, Section 2.3 describes the vehicle control of within the topic of autonomous racing.

2.1 Autonomous Vehicle System

Problems in finding collision-free trajectories do not limit themselves in the automotive field, but also exist in fields like nautics, aerospace and robotics. Although path planning for robots and vehicles are very much alike, there are some features responsible for some crucial differences between the two. Therefore, methods that are applied to robots commonly cannot be employed directly for autonomous vehicles. Complex driving situations for autonomous vehicles develop a need for adaptation of the method to these complex driving situations. It is simply not feasible to predefine all possible all possible traffic situations that can be encountered.

One big difference that needs to be addressed is those of a robot being holonomic or non-holonomic. A car is an example of a non-holonomic vehicle, indicating that it only can move forward/backward in the direction perpendicular to the rear wheel axis. This means that a car is, in contrary to holonomic (i.e. omniwheel) robots, unable to move straight to the side. Moreover, the turning radius of a car is limited due to mechanical bounds on the wheel. Taking non-holonomic constraints as such into account, makes it a lot harder to plan- and track trajectories for non-holonmic robots than omniwheel robots.

Recall from Chapter 1 that roughly perception, planning and control are distinguished in the software system of an autonomous vehicle. Within these components, studies in autonomous behavior have been going on for years. Beyond the regular speed regulation, like Adaptive Cruise Control (ACC) applications, autonomous behavior like car following or platooning all contribute to the autonomous behavior of future cars (Ozguner, Acarman, and Redmill, 2011). Figure 2.1 depicts the autonomous vehicle system and its interconnecting parts as a whole.

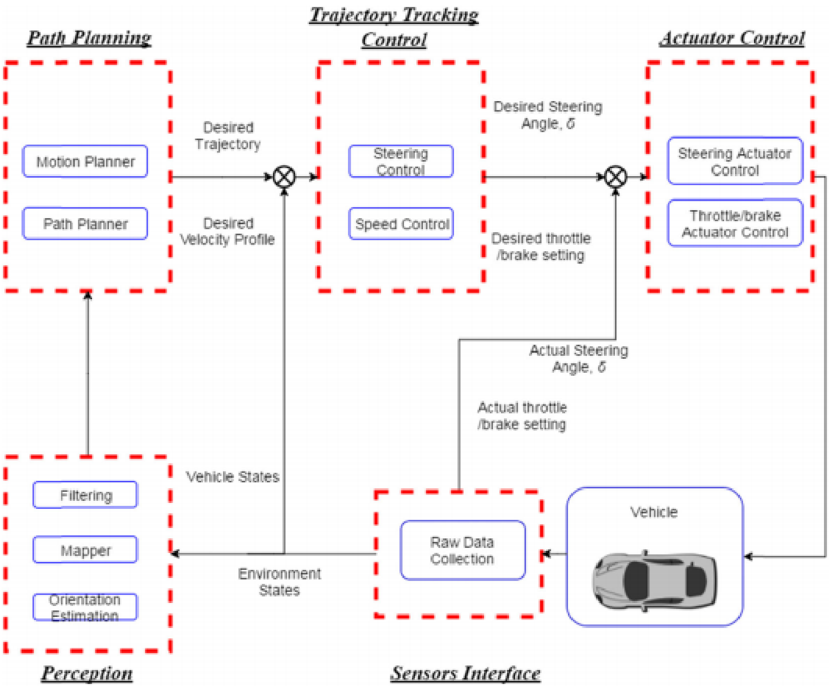


FIGURE 2.1: Overview of autonomous vehicle system (Amer et al., 2017)

2.1.1 Sensors

As briefly stated before, sensing and perception have the important task of gathering information about the surrounding environment to recognize specific objects. Sensors are the components in the car that enables this, just as eyes for a human being enable to understand what exactly is going on in the surroundings. In terms of obstacle avoidance in path planning, many different types of sensors haven been development, amongst which sonar, infrared/ultrasonic sensors, LiDAR, laser range finders and camera (Anavatti, Francis, and Garratt, 2015). From these sensors types, data processing techniques are applied to update positions and directions of the moving obstacles. Moreover, it determines current states such as the speed and heading angle of the controlled vehicle. Each type of sensor have their benefits and disadvantages and for this reason, most autonomous vehicles use a combination multiple sensors to add extra redundancy. Combining multiple sensor readings compensates for the weaknesses of various sensors is often referred to as sensor fusion.

2.1.2 Vehicle Overtaking Manoeuvre

When considering a certain path or route that a vehicle has to take, one can imagine that several maneuvers (e.g. lane changes, crossing intersections or left/right turns) have to be taken in order to successfully reach the destination point. Dealing with maneuvers like these is also an important aspect of path planning and trajectory control. (Zhang et al., 2013) state that, when driving in a structured environment, complex maneuvers can be decomposed into two basic sub-maneuvers: lane changing and lane-keeping. An overtaking manoeuvre is an excellent example for this statement, since it can be decomposed in a series of sub-maneuvers. Considering a high-way situation where a faster car want to overtake a slower car, then the actual overtake manoeuvre consist of respectively a left lane change, lane keeping and a right lane change. Figure 2.2 depicts this manoeuvre, as *(i)* represents the left lane change, *(ii)* the lane keeping manoeuvre and *(iii$_a$)* the right lane change. The subject vehicle (SV) overtakes the leading vehicle (LV), while maintaining a safe lateral, approaching and merging distance. Although Figure 2.2 shows an overtaking manoeuvre on a straight path where left-side overtaking is often only included, research on possible overtaking on the 'wrong' side is also done in (Kala and Warwick, 2015).

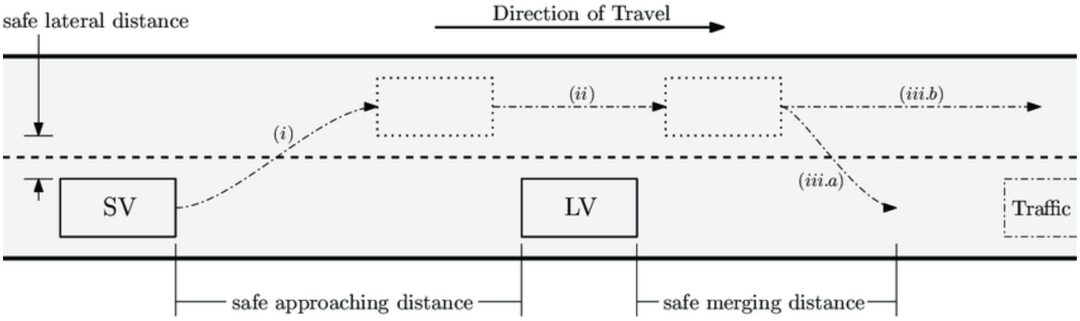


FIGURE 2.2: Overview of a basic overtaking manoeuvre

2.2 Vehicle Model Strategies

The way in which the vehicle is modelled plays a crucial role in two aspects of the path planning & trajectory control. (Amer et al., 2017) reviewed the state of the art modelling techniques in path tracking control, from which the highlights are analysed in this section. Several types of vehicles models, depending on its linearity and type of behavior it simulates, can be categorised. Vehicle models play a crucial role in two aspects of the controller development. The first aspect is the simulation of the vehicle system. During the development of any controller, a vehicle model is usually employed to simulate the vehicles behaviour under the influence of the proposed controller. The second aspect concerned with the vehicle model is the control law in path planning and trajectory control, which is often based on the mathematical representation of the vehicle system. The most common way to do so, is by employing a linearized vehicle model. However, several promising developments in establishing controllers based on non-linear vehicle models have been proposed as well. In general, the types of modelling are categorised in geometric, kinematic and dynamic vehicle models. Each configuration has its own advantages and purpose, depending on the implementation and properties that are desired to be studied. To give some perspective on the common representation of a vehicle model generally used in handling studies, the vehicle axis system with common nomenclature and important symbols is given in Figure 2.3.

The roll, pitch and yaw axes coincide at the vehicle's center of gravity (CG). The variables corresponding to the different axes will contribute in the path planning and trajectory control of the study. To what extend each variable will play a role depends on the particular vehicle model considered and is elaborated on in the next sections.

2.2.1 Geometric

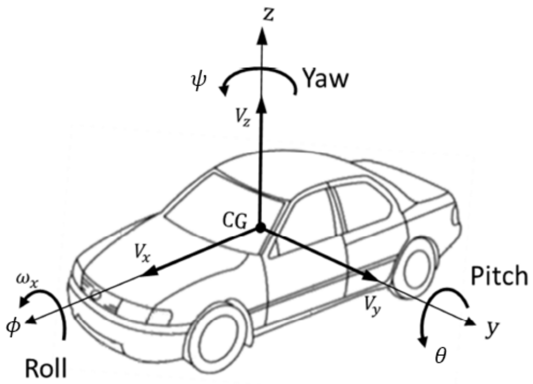


FIGURE 2.3: Vehicle Axis System used in Vehicle Modelling (Kissai et al., 2019)

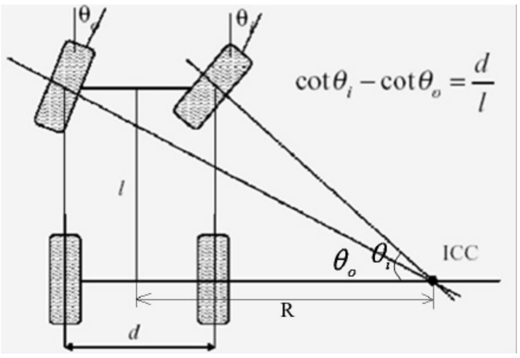


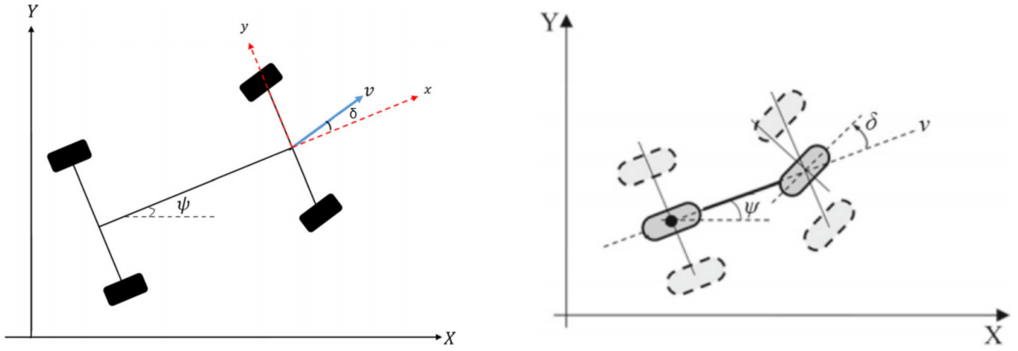
FIGURE 2.4: Ackermann steering configuration (Amer et al., 2017)

In the field of trajectory control, deriving control laws via a geometric approach is repeatedly proposed in literature (Lee, Leok, and McClamroch, 2010), (Gajbhiye and Banavar, 2016). With regard to autonomous vehicles, the geometric vehicle model is often used in path planning and trajectory control. By modelling a vehicle in a geometric manner, only geometrical dimensions of the vehicle are considered and kinematic and dynamic properties are neglected. During any manoeuvre, this model only considers the dimensions and positions of the vehicle with no regard to its velocity and acceleration. It is constructed based on the Ackermann steering principle, which states that the line perpendicular to all wheels, should intersect at an imaginary point in the extension of the rear axis (Figure 2.4). This model was especially important in developing the pure pursuit method, one of the most popular trajectory tracking controllers (Amer et al., 2017). Due its simplicity in parameters, this model is hardly ever used for simulation of the vehicle state. The most important parameter in this model is the steering angle, which can be derived from Figure 2.4 as

$$\theta = \tan^{-1}\left(\frac{l}{R}\right) \tag{2.1}$$

2.2.2 Kinematic

Kinematic can be defined as the study of a motion of a body regardless of that body's internal forces, inertia and energy. The corresponding kinematic vehicle model, therefore, describes the motion of the vehicle in terms of its position, velocity and acceleration without any regard of its internal dynamics. The kinematic vehicle model is widely used in studies regarding path planning and trajectory control, due to its simplicity and its feature to be able to describe the motion of the vehicle. Important aspects like vehicle velocity, acceleration in lateral direction and its yaw motion with respect to fixed axes (or local coordinates) and global axes (global coordinates) are described by the kinematic vehicle model. This vehicle is also able to both simulate the vehicles behavior like position, velocity and acceleration during manoeuvring, as well as deriving the mathematical representation used for the control law of the system. Kinematic vehicle models are often employed for control at relatively low speeds, since dynamics are not included (Schwarting, Alonso-Mora, and Rus, 2018). These types of models can be roughly decomposed in three different types of models; the full vehicle model (Figure 2.5a), the bicycle model (Figure 2.5b) and the extended bicycle model including slip angle.



(A) Full vehicle model

(B) Reduced bicycle model

FIGURE 2.5: Kinematic vehicle models

Full Vehicle Kinematic Model

This model represents the vehicle kinematics by perceiving the car as a whole and describes it by means of two sets of coordinates. Firstly, the local coordinates (x, y) are set to act as rigid body of the vehicle. Secondly, global coordinates (X, Y) are set to be fixed on the earth coordinates. Since this research considers an overtaking manoeuvre, the considered vehicle is always moving and the local coordinates are, therefore, always taken as moving coordinates with respect to the global coordinates. Considering the kinematic full vehicle model in Figure 2.5a, the following kinematic equation can be formulated:

$$\begin{bmatrix} v_X \\ v_Y \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{2.2}$$

where $v_X = v\cos\psi$ and $v_Y = v\sin\psi$.

Kinematic Bicycle Model

A simplified version of the full vehicle and the four wheels are reduced to only two wheels is the kinematic bicycle model, as can be seen in Figure 2.5b. Despite the fact that the model is less complex, different studies have shown that the kinematic bicycle model for path planning and trajectory control is sufficient for generating satisfactory results (Liu et al. (2014), Kong et al. (2015)). The biggest advantage of using the kinematic bicycle model is the permitted neglect of the slip of each wheel to the vehicle's direction by assuming only one wheel per axis. Consequently, the driving- and velocity direction of the vehicle are equivalent.

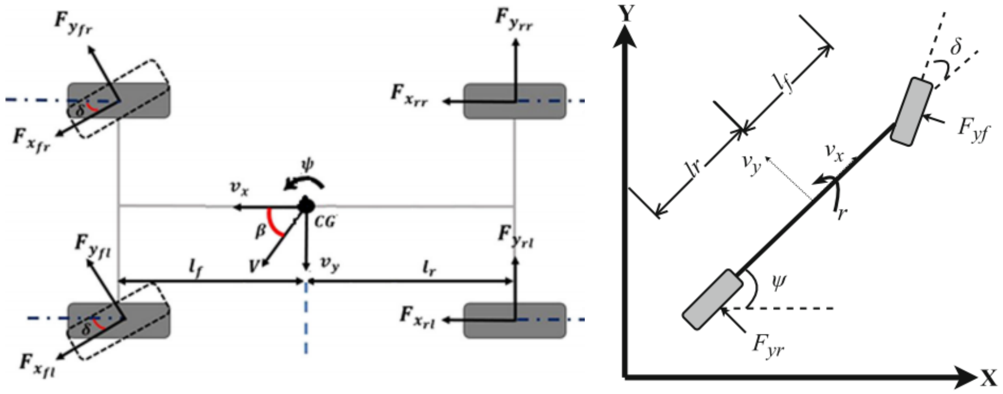
Extended Kinematic Bicycle Model

Some studies (Schwarting et al. (2017), Lucet, Lenain, and Grand (2015)) use a extended version of the kinematic bicycle model to include the slip angle on both rear- and front wheel to allow for slippery terrains. Its graphical form stays intact, but the kinematic equations will differ since new slip velocities are introduced. The new symbols V_{LR} and V_{SR} respectively express the rear wheel longitudinal slip velocities and the real wheel side slip velocities. β_R refers to the rear wheel sideslip angle and β_F is the sideslip angle for the front wheel. Considering these extra entities, the following equations of motion can be derived:

$$\begin{aligned} v_X &= v\cos\psi - V_{LR}\cos\psi - V_{SR}\sin\psi \\ v_Y &= v\sin\psi - V_{LR}\sin\psi + V_{SR}\cos\psi \\ \dot{\psi} &= \tfrac{v-V_{LR}}{L}\left[\tan\beta_R + \tan\left(\delta-\beta_F\right)\right] \end{aligned} \tag{2.3}$$

2.2.3 Dynamic

An even more complex way of modelling a vehicle is to include its dynamics: the internal forces, energy or momentum. While considering these elements, the dynamic vehicle model describes the motion of the vehicle in terms of its position, velocity and acceleration. In contrary to the kinematic vehicle model, the dynamic version includes tire forces and, therefore, perform better in studies that consider vehicles operating at high speeds or performing aggressive manoeuvres (Schwarting, Alonso-Mora, and Rus, 2018). Besides that, the overall geometric and kinematic relationships discussed previously are included as well, resulting in an exhaustive vehicle model. Recall from Section 2.2 that linearity play a role in categorising the vehicle models as well. Due to the added internal forces, energy and momentum, the dynamic vehicle model is subdivided into two main categories: linear and non-linear dynamic models. For each of these classes, again the full vehicle- and bicycle model are distinguished and are depicted in respectively Figure 2.6a and Figure 2.6b.



(A) Full vehicle model

(B) Reduced bicycle model

FIGURE 2.6: Dynamic vehicle models

Linear Dynamic Models

A common method to derive the equations of motion in the dynamics of a vehicle is using the Newtonian approach. For the longitudinal plane, this method includes translational motion in lateral x, longitudinal y and rotational motion around the z-axis. Recall from Figure 2.3 that in vehicle dynamics these motions around the z-axis is better known as yaw ψ and yaw rate r.

The equations of motion of the dynamic full vehicle model are written in Eq. 2.4, based on (Junmin Wang, Steiber, and Surampudi, 2008).

$$\begin{aligned}
&F_{xrr} + F_{xrl} + F_{xfl}\cos\delta + F_{yfl}\sin\delta + F_{xfr}\cos\delta + F_{yfr}\sin\delta = m_b a_x \\
&F_{yrr} + F_{yrl} - F_{xfl}\sin\delta - F_{xf}\sin\delta + F_{yfl}\cos\delta + F_{xf}\cos\delta + F_{yfr}\cos\delta = m_b a_y \\
&\sum M_{zij} + \Big[-F_{yrr} - F_{yrl}\Big]\, l_r + \Big[F_{xfl}\sin\delta + F_{yfl}\cos\delta + F_{xfr}\sin\delta + F_{yfr}\cos\delta\Big]\, l_f \\
&+ \Big[F_{xfr}\cos\delta - F_{xfl}\sin\delta + F_{yfl}\sin\delta - F_{yfr}\sin\delta - F_{xrl} + F_{xrr}\Big]\, \tfrac{t}{2} = I_{z,CG}\ddot{\psi}
\end{aligned} \tag{2.4}$$

Following Eq. 2.4, the vehicle is considered to have the following elements:

- sprung mass m_b
- moment of inertia of the sprung mass about the z-axis I_{CG}
- acceleration a_x of longitudinal motion in x-direction
- acceleration a_y of lateral motion in y-direction
- yaw motion with yaw angle ψ about the z-axis

The position of each wheel of the vehicle is indicated with either subscript i or subscript j, where i stand for front or rear and j denotes left or right. By doing so, all wheels can be denoted separately. This is not necessary when the bicycle model is used, since it only considers a front and rear wheel. The other elements are, however, used in the simplified dynamical bicycle model and the corresponding equations of motion, based on (Hoffmann et al., 2007), are shown in Eq. 2.5. The tire forces are generated due to interaction of the tires with the surface. During manoeuvring, the tires deform in both longitudinal and lateral direction causing friction forces which are originally non-linear. For this reason, linear dynamic models commonly employ linearization by assuming linear tire forces acting on each of the tires, i.e. a linear tire model. These types of tire models assume a linear relationship between slip angles and resp. both lateral and longitudinal tires forces, F_y and F_x.

$$\begin{aligned}
&F_{xr} + F_{xf}\cos\delta + F_{yf}\sin\delta = m_b a_x \\
&F_{yr} - F_{xf}\sin\delta + F_{yf}\cos\delta = m_b a_y \\
&F_{yr} l_r + \Big[F_{xf}\sin\delta + F_{yf}\cos\delta\Big]\, l_f = I_{z,CG}\ddot{\psi}
\end{aligned} \tag{2.5}$$

Considering both Eq. 2.4 and Eq. 2.5, it can be noticed that the tire forces are the main contribution in the dynamic vehicle model. It is the main contributing external factor and it can also be considered as main source of external disturbance as well as the grip that governs the vehicle's motion.

Non-linear Dynamic Models

When a non-linear vehicle model is used in deriving the control law of the subject vehicle, tire models are also employed to simulate the lateral and longitudinal tire forces on each separate wheel during any manoeuvre. Considering non-linearity of tire dynamics enables to simulate the vehicle's response in high-speed and large steering angle situations. Moreover, including non-linearity of tires can improve the controller performance when considering a vehicle at the limit of friction. Incorporating a non-linear tire model into the controller design is one of the approaches to include tire non-linearity and is suitable for both the full vehicle- and bicycle model.

The choice of tire model depends on how the vehicle dynamics are structured. A suitable tire model for handling non-linear tire dynamics is examined in the next section.

2.3 Path Planning & Trajectory Control at the limit of friction

Trajectory control is a crucial function of an autonomous overtaking system and a variety of control methods have been proposed in current literature. Path planning and trajectory tracking constitute the two main functions of this particular control field and for these functionalities a collection of state-of-the art strategies is given in (Dixit et al., 2018). Figure 2.7 presents the general control architecture for autonomous vehicles and constitute, as stated, of path planning (considered the same as trajectory planning) and trajectory tracking.

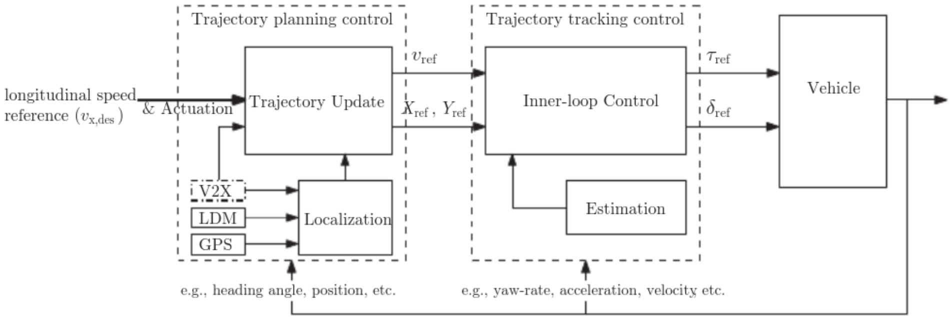

Trajectory planning control
Trajectory tracking control
longitudinal speed reference ($v_{x,des}$)
& Actuation
Trajectory Update
v_{ref}
X_{ref}, Y_{ref}
Inner-loop Control
τ_{ref}
δ_{ref}
Vehicle
V2X
LDM
GPS
Localization
Estimation
e.g., heading angle, position, etc.
e.g., yaw-rate, acceleration, velocity etc.

FIGURE 2.7: General control architecture for an autonomous vehicle

In general, there are two basic approaches to trajectory generation with known path information. The first one is to optimize both trajectory generation and tracking simultaneously, while the second one decouples both. Since MPC is known to be suitable for both path planning and trajectory tracking, the integrated approach is further examined in this section. MPC has become a popular method in vehicle control and (Guo et al., 2018) use it to design a simultaneous trajectory planning and tracking controller for an intelligent vehicle. According to (Schwarting et al., 2017), MPC performs well in complex, high-speed situation where full dynamic vehicle models are employed. Considering that the vehicle operates at the limit of friction, a dynamic vehicle model is essential to include tire friction forces. (Kritayakirana and Gerdes, 2012) were the first to introduce a concept called Centre of Percussion (COP) in vehicle control by applying to design a steering controller for an autonomous race car, to minimize both the heading and lateral deviations. Since then several researchers applied COP and utilized it as error reference point in simulations, leading to simplified calculations and improved stability for vehicle operating at the limit of friction. The COP is the point on a certain object at which translational and longitudinal forces cancel each other out. At this point, called a pivot point, the rear tire forces do not influence the lateral motion of the COP. (Hiraoka, Nishihara, and Kumamoto, 2009) used the COP to propose an automatic path-tracking controller of a four-wheel steering (4WS) vehicle based on the sliding mode control theory, where the centres of percussion with respect to the rear/front wheels are front/rear control points of a full dynamic vehicle model. In (Song et al., 2015), COP is used in the controller by utilizing a dynamic bicycle model. Since the rear lateral force produces a lateral and a rotational component at the centre of mass, the rear tires do not exert

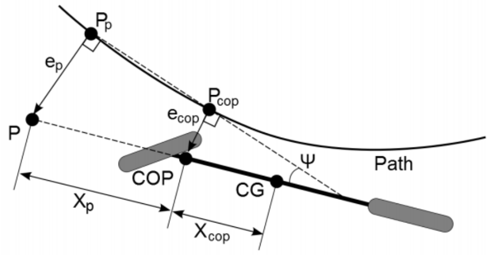


FIGURE 2.8: Effect of using Centre of Percussion on a dynamic bicycle model

a lateral force at the COP and the lateral effects of both components are cancelled out at the COP. This phenomenon is illustrated by means of Eq. 2.6 and Figure 2.8.

$$\begin{aligned} x_{cop} &= \frac{I_z}{l_r m} \\ \frac{F_{yr}}{m} &= x_{cop}\frac{l_r F_{yr}}{I_z} \end{aligned} \tag{2.6}$$

I_z is the moment of inertia around the z-axis, l_r is the distance between the rear axle and the center of mass.

2.3.1 Racing line

Within the formula 1, a racing line is referred to as the fastest line or arc through a corner on a race circuit. The trajectory of the racing line depends on the severity of the corner, how long the following straight is and what kind of car is being driven. The goal is to always carry as much speed in the braking zone, through the corner and onto the next straight.

In most cases, the racing line makes use of the entire width of the track to lengthen the radius of a turn. This means entering at the outside edge, touching the apex-point on the inside edge, whereafter it exits the turn by returning outside.

Figure 2.9 shows the fastest race line on a 90 degrees curve. The apex is a point on the inside edge of a curve that allows the vehicle to have the smallest curvature and, consequently, the largest speed possible when cornering.

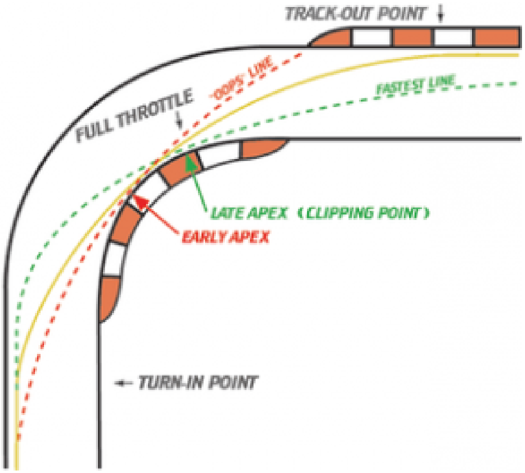


FIGURE 2.9: Perfect racing line

Chapter 3

Vehicle Model

The vehicle model plays a crucial role in designing the controller, elaborated on in the next chapter, for two main reasons. First, the derivation of the control law is often based on the mathematical representation of the vehicle model. Second, the vehicle model is commonly used to simulate the behaviour of the proposed vehicle model and the performance of the proposed controller. The employed vehicle model for this thesis, the reasoning of this choice and its benefits and drawbacks are explained in Section 3.1. Subsequently, Section 3.2 discusses the reformulation of the vehicle dynamics in terms of the center line of the track, represented by a curvilinear coordinate system.

3.1 Kinematic Bicycle Model

As explained in Chapter 2, dynamic vehicle models tend to perform better in high-speed situations than kinematic vehicle models. Especially for racing situations where friction limit at the tires is considered, tire forces should be included. For these reasons, a full dynamical vehicle model would be beneficial to derive the equations of motion used in the controller design.

However, using high-complex vehicle models lead to extremely difficult computations often yielding high and inefficient computation times. As explained in Chapter 2, the simpler models like the kinematic bicycle model have been proven to show quite some resemblance in terms of trajectory planning performance, regardless of the simplifications made in the model. The biggest advantage of the bicycle model is the permitted neglect of slip for each wheel, since the wheels are simplified to be attached to only one frame. Hence, driving- and velocity direction of the vehicle are the same. For the sake of the mentioned simplicity and assuming that the model is accurate enough, the kinematic bicycle model is used to generate the optimized trajectories in this thesis. Moreover, (Kong et al., 2015) have shown that the accuracy of the bicycle model can be improved by increasing the discretization time of the model. For this thesis, that is beneficial for the discretization step length, as a larger step length allows for planning further ahead in the track. Section 3.2 describes the reformulation of the time-domain kinematic equations to space-based equations and its consequent advantages.

The assumption that slip for each tire in the kinematic bicycle model is eliminated, is related to the non-holonomic nature of the car-like robot. Non-holonomic means that the vehicle has two degrees of freedom, in this case forward/backward motion and turning. This is obvious the case for any car-like vehicle. According to (De Luca, Oriolo, and Samson, 1998), a Rear-Wheel Driven (RWD) non-holonomic vehicle's movements can approximately be described by its time-domain kinematic equations. Given the fact that formula 1 cars are RWD and non-holonomic by nature, it can be modeled as the aforementioned kinematic bicycle model as follows:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\psi} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} \cos\psi \\ \sin\psi \\ \tan\delta/L \\ 0 \end{bmatrix} v_1 + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_2 \tag{3.1}$$

where $\dot{x}, \dot{y}$ are the time-derivatives of the vehicle's coordinates (x, y) in the global coordinate system, ψ the yaw angle, δ the steering angle and L the wheel-to-wheel distance of the vehicle. v_1 and v_2 denote the driving- and steering input, respectively. Modelling a car-like robot like this comes with one singularity in the lateral domain. When the front wheels have turned 90 degrees ($\delta = \pm\frac{\pi}{2}$) normal to the longitudinal axis of the chassis, the car becomes jam. However, this discontinuity is often resolved by setting constraints on the steering angle.

Considering that the slip angles at both tires of the bicycle model are zero, leads to elimination of the lateral component of the vehicle velocity ($v_1 = v_x$). This gives rise to equations of motions purely based on geometric relationships governing the system. Hence, (3.1) can be reformulated as the following set of differential equations:

$$\begin{cases} \dot{x} = \dfrac{\mathrm{d}x}{\mathrm{d}t} = v_x \cos(\psi) & \text{(3.2a)} \\ \dot{y} = \dfrac{\mathrm{d}y}{\mathrm{d}t} = v_x \sin(\psi) & \text{(3.2b)} \\ \dot{\psi} = \dfrac{\mathrm{d}\psi}{\mathrm{d}t} = \dfrac{v_x}{L} \tan(\delta) & \text{(3.2c)} \end{cases}$$

As stated before, it is assumed that the wheels do not slip at surface contact, but can freely rotate around their rotation axis. Vehicle control mainly focuses on the lateral and longitudinal aspect of the utonomous vehicle. This often entails in the steering angle δ being the control input for the lateral domain and acceleration as control input for the longitudinal domain. The steering angle can directly be related to the vehicle curvature κ_v as:

$$\kappa_v = \frac{\tan(\delta)}{L}.$$

Implementing this into Eq. (3.2), extending the model with the longitudinal acceleration and taking the derivative of vehicle curvature gives the following equations:

$$\dot{\psi} = \kappa_V v_x$$

$$\dot{v_x} = -\mu v_x + a_x$$

$$\dot{\kappa}_v = C_v$$

where a_x is the longitudinal acceleration, μ a longitudinal friction coefficient and C_v the curvature rate of the vehicle. The influence of μ on the trajectories is investigated during the simulation phase.

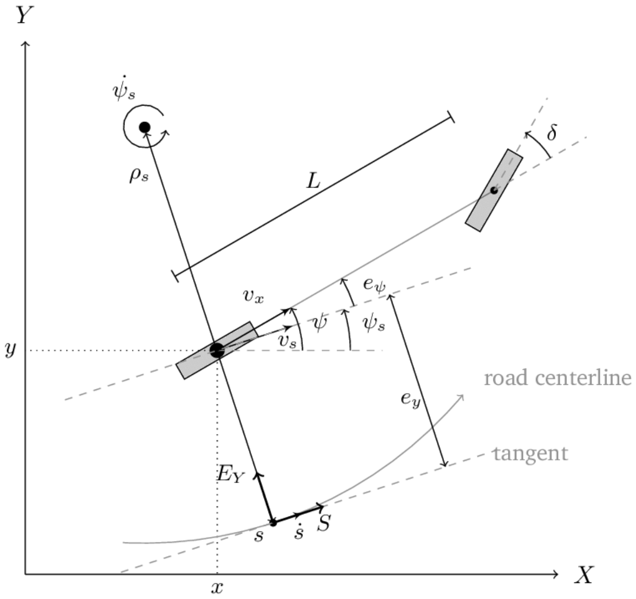


FIGURE 3.1: Considered kinematic bicycle model in global- and curvilinear coordinate system

The main contribution of the curvilinear coordinate system is the introduction of the vehicle's position along the center-line of the track as variable s. Furthermore, the lateral deviation and the heading displacement between the vehicle and the center-line of track are respectively modelled as E_y and E_ψ. The way of deriving these variables in done geometrically and is explained in the subsequent section.

The time-domain kinematics as explained can be formulated as system of Ordinary Differential Equations (ODE) as follows:

$$\dot{\zeta}(t) = f(\zeta(t), u(t)) \tag{3.3}$$

where $\zeta(t) \in \mathbb{R}^n$ is the state of the system at time t, $u(t) \in \mathbb{R}^m$ the control inputs, n the number of states and m the number of control inputs. The states in the time-domain are defined above as follows: $\zeta = [E_y, E_\psi, v_x, \kappa_v]^T$. The two control inputs are $u = [a_x, C_v]^T$. Note that these states are defined for the curvilinear coordinate system, as the global coordinates x and y are changed to E_y and E_ψ The considered vehicle model is given in Figure 3.1, illustrating the relationship between two different coordinate systems considered in this thesis: the global coordinate system and the curvilinear coordinate system.

3.2 Spatial Dynamics Reformulation

Instead of commonly used time-dependent dynamics, track-dependent (or spatial) dynamics are proposed in this thesis. By doing so, a natural formulation of obstacles and road bounds under varying vehicle speed can be formulated (Buyval et al., 2017). Moreover, describing the dynamics in the space-domain has the convenient advantage of making the system independent of both time and speed. Consequently, track limitations become simple (convex) state bounds, which are independent of the vehicle speed. The main detail of formulating the dynamics in this way, is that the dynamics are derived about a curve σ that defines the center-line of the track. It is assumed that track information is known beforehand, since these parameters are necessary in the computations of the vehicle dynamics. Figure 3.1 describes the kinematic bicycle model in both frames, where the coordinate s defines arc-length along the track σ as the projected vehicle position along the track. The derived spatial coordinates E_y and E_ψ, critical in the curvilinear coordinate system, are shown. The rest of the spatial states are given in Table 3.1 and how they are derived is explained below.

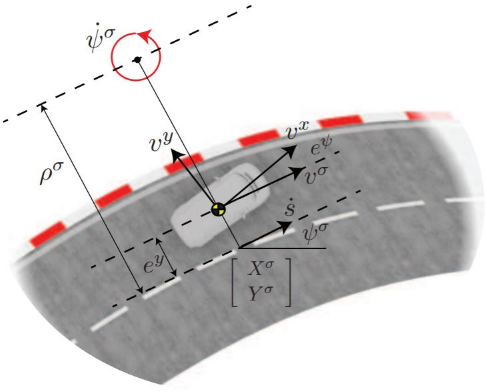


FIGURE 3.2: Visualization of inertial coordinates

From Figures 3.1 and 3.2, important kinematic equations can be derived geometrically as follows. The lateral displacement of the vehicle with respect to the center-line is

$$E_y := \left\| \begin{bmatrix} X \\ Y \end{bmatrix} - \begin{bmatrix} X^\sigma \\ Y^\sigma \end{bmatrix} \right\| \tag{3.4}$$

and the yaw angle of the vehicle with respect to the path is

$$E_\psi := \psi - \psi^\sigma, \tag{3.5}$$

where ψ^σ in Figure 3.2 corresponds to ψ_s in Figure 3.1. X^σ and Y^σ denote the position of the current reference point on the path s and ψ^σ the orientation. Other important geometric relations, essential for the formulation of the vehicle dynamics in this thesis, can be derived from Figures 3.1 and 3.2 and are given below:

$$v_\sigma = (\rho^\sigma - E_y)\dot{\psi}^\sigma = v_x \cos(E_\psi) \tag{3.6}$$

$$\dot{s} = \rho^\sigma \dot{\psi}^\sigma = \frac{\rho^\sigma}{\rho^\sigma - E_y} v_x \cos(E_\psi) = \frac{v_x \cos(E_\psi)}{1 - E_y \kappa_s} \tag{3.7}$$

where v_σ is the projected vehicle velocity along the tangent (i.e. the direction) of the path, ρ^σ the radius of the local curvature of σ, $\dot{\psi}^\sigma$ the rate of change of the path orientation and $\dot{s}$ is the vehicle velocity along the path (note the difference between v_σ and $\dot{s}$). Recall from Section 3.1 the variables E_y and E_ψ, indicating the position of the vehicle relative to the center-line of the track in the curvilinear coordinate system. From Figures 3.1 and 3.2, the time derivatives of these spatial coordinates are derived geometrically as well

$$\begin{aligned} \dot{E}_y &= v_x \sin\left(E_\psi\right) \\ \dot{E}_\psi &= \dot{\psi} - \dot{\psi}^\sigma \end{aligned} \tag{3.8}$$

The spatial dynamics of state vector ζ in relation to the time-dependent dynamics are given following the chain rule expressing the spatial derivative ($\zeta^{'}$) as function of the time derivative ($\dot{\zeta}$), following (Frasch et al., 2013):

$$\zeta^{'}(s) := \frac{d\zeta}{ds} = \frac{d\zeta}{dt}\frac{dt}{ds}. \tag{3.9}$$

Assuming that $\dot{s} \neq 0$ at all times, we get $\frac{dt}{ds} = \frac{1}{\dot{s}}$. Therefore, the spatial states can be derived generically by expressing it in terms of $\dot{s}$ as follows:

$$\zeta'(s) = \dot{\zeta}\frac{1}{\dot{s}} \tag{3.10}$$

where $\dot{\zeta}$ is defined as in Eq. (3.3). Considering this reformulation of dynamics, the final spatial vehicle dynamics can be derived according Eq. (3.10):

$$E_y'(s) = \frac{\dot{E}_y}{\dot{s}} = (1 - E_y\kappa_s)\tan\left(E_\psi\right) \tag{3.11a}$$

$$E_\psi'(s) = \frac{\dot{E}_\psi}{\dot{s}} = \frac{\left(1 - E_y\kappa_s\right)}{\cos\left(E_\psi\right)}\kappa_v - \psi'^{\sigma} \tag{3.11b}$$

$$v_x'(s) = \frac{\dot{v}_x}{\dot{s}} = -\frac{\left(1 - E_y\kappa_s\right)}{\cos\left(E_\psi\right)}\mu + \frac{\left(1 - E_y\kappa_s\right)}{v_x\cos\left(E_\psi\right)}a_x \tag{3.11c}$$

$$\kappa_v'(s) = \frac{\dot{\kappa}_v}{\dot{s}} = \frac{\left(1 - E_y\kappa_s\right)}{v_x\cos\left(E_\psi\right)}C_v. \tag{3.11d}$$

Equations (3.11a) – (3.11d) describe the spatial dynamics considered for the kinematic bicycle model. These dynamics are considered throughout the thesis and used for the MPC formulation in the controller design. The complete reformulation from time (Eq. (3.3)) to spatial vehicle dynamics can now be formulated as:

$$\zeta'(s) = f(\zeta(s), u(s)), \tag{3.13}$$

where the corresponding states and control inputs are given in respectively table 3.1 and 3.2. Note that the original control inputs are given as acceleration a_x and car curvature κ_v. However, in Eq. (3.11c) κ_v is represented as state the inputs for the vehicle model are a_x and C_v. To ensure an accurate model discretization, where the rates of vehicle inputs are constant between samples, the same must hold for the control inputs between samples.

Consequently, by considering the kinematic bicycle model within the curvilinear coordinate system, all computations are done within this frame as well. Since distance and time within this topic are significantly correlated, time information can - if necessary - be retrieved by integrating $\frac{dt}{ds}$ along σ as in Eq. (3.14):

State	Unit	Description
E_y	m	Deviation from center-line
E_ψ	rad	Yaw angle relative to path
v_x	m/s	Longitudinal velocity
κ_v	m^{-1}	Vehicle curvature

TABLE 3.1: Spatial States of the vehicle model

Control	Range	Unit	Description
a_x	[-5, 5]	$\mathrm{m/s}^2$	Longitudinal acceleration
C_v	[-0.2, 0.2]	m^{-2}	Vehicle sharpness

TABLE 3.2: Control inputs of the spatial vehicle model

$$t(s) = \int_{s_0}^{s} \frac{1}{\dot{s}(\tau)} \mathrm{d}\tau \tag{3.14}$$

As stated, the computations are done in the spatial domain. However, inertial coordinates may be recovered by transforming the spatial coordinates back to the Cartesian frame as follows:

$$\begin{aligned} X &= X^\sigma - E^y \sin\left(\psi^\sigma\right) \\ Y &= Y^\sigma + E^y \cos\left(\psi^\sigma\right) \\ \psi &= \psi^\sigma + E^\psi \end{aligned} \tag{3.15}$$

This is essential for the MPC controller explained in the next chapter. The reference trajectories are calculated in the spatial domain, visualized in the global frame by using Eq. (3.15) and derived back to the curvilinear coordinate system again. This is an iterative process, for which the amount of iterations depend on the length of the track segment. The exact working method of the algorithm is given by means of a pseudo code in section 4.4.

Chapter 4

Controller Design

The proposed vehicle model of the previous chapter has, as explained, influence on the way the controller is designed. This chapter elaborates on the aspects belonging to the design of the controller. In section 4.1, the choice of using Model Predictive Control will be clarified and the method at hand will be described. The reasoning behind the objective of the controller is explained in section 4.3. Section 4.3 describes the derivation of the successive linearized an discretized vehicle dynamics. The final control structure and MPC formulation, including made assumptions led to this formulation, are given in section 4.4.

4.1 Model Predictive Control (MPC)

In terms of vehicle control, dynamic controllers tend to perform better in complex driving situations since vehicle dynamics are taken into account. This is done either by directly deriving a control law from the dynamical vehicle model or by including some of the dynamical properties. Model Predictive Control (MPC) combines the properties of dynamic, optimal and adaptive controllers resulting in a suitable approach to tackle the problem at hand. MPC is also known as Receding-Horizon (RC) and owes its success to its simplicity in terms of logic, making it well suitable in handling non-linearities and constraints (Allgöwer and Zheng, 2012). MPC uses a model to predict future states of the system. The trajectory planning and execution are performed at each step, allowing to take both the car dynamics and the position of other vehicles into account. The term MPC does not designate a specific control strategy to the vehicle model, but rather uses an abundant range of control methods, which make explicit use of the model at hand. By doing so, optimal control inputs to the model are determined, based on minimization of a certain objective function. The objective function is subjected to certain constraints that set physical limits on the vehicle model, such as maximum acceleration, but the constraints are also used to formulate limitations in the track environment.

The main challenge of using MPC formulation for optimization problems, is the guarantee of real-time feasibility and convergence. Typically, when formulating the problem in a nonlinear and non-convex framework, this appears to be extremely difficult and not feasible ((Frasch et al., 2013)). For this reason, the dynamic equations of the vehicle are successively linearized and discretized (see section 4.3), allowing the problem to be formulated as a convex optimization problem.

A path planning problem can also be formulated as a convex optimization problem by linearizing the vehicle dynamics and constraints (Bevan, Gollee, and O'reilly, 2010). Bevan et al. (2010) enforce the vehicle to be within the space-varying road boundaries by taking vehicle dynamics explicitly into account during the path planning phase. The spatial kinematic bicycle model in combination with the curvilinear coordinate system, as explained in 3 , is used to predict the motion of the simulated vehicle (and controlling it in a receding horizon manner).

In this case, the number of points in the prediction horizon and the discretization step length are determined after several experiments with varying horizon length and step size. Important is that the prediction horizon is long enough to optimize over the subsequent turn, since these are the track segments we are interested in. In general, a shorter discretization step is beneficial in terms of model prediction accuracy, since the control input is not constant for so long. However, a shorter discretization step would require more prediction horizon points, which would lead to a more expensive optimization problem. This is a common trade-off within the MPC method and is elaborated on further in chapter 6.

4.2 Control Ojective

4.2.1 Racing Behaviour

With autonomous racing being the overarching subject in this thesis, the MPC controller aims at solving an optimization problem suitable and reasonable for racing behaviour. This is one of the reasons for choosing to express the vehicle dynamics in terms of spatial coordinate s. In the most rational way of thinking, racing is to get to finish line as fast as possible. In other words, the time to travel from starting- to finish line must be minimized. Considering the explained dynamics in Chapter 3, minimizing the travel time can be expressed in terms of progression along the center-line of the track as maximizing $\dot{s}$, inspired by (Lima et al., 2018). Since minimization problems are more common in practice, the problem will be denoted as minimization of $\frac{1}{\dot{s}}$. Considering Eq. (3.7) and implementing the choice of $\frac{1}{\dot{s}}$ results in minimizing, as explained in section 4.1, the following Receding-Horizon objective function:

$$\frac{1}{\dot{s}} = \frac{1 - \kappa_s E_y}{v_x \cos(E_\psi)} \tag{4.1}$$

As explained, convex optimization problems tend to give results representative for real-time implementation. For this reason, $\frac{1}{\dot{s}}$ is approximated by Taylor expansions, imposing a linear (and therefore convex) objective function. Proof of convexity of linear functions is given in Appendix A. Following Appendix B.1, $\frac{1}{\dot{s}}$ is approximated by Taylor approximations with respect to $E_{y,ref} \neq 0$, $E_{\psi,ref} = 0$ and $v_{x,ref} \neq 0$ resulting in

$$\frac{d\frac{1}{s'}}{dt} = \gamma_1 E_y + \gamma_2 E_\psi^2 + \gamma_3 v_x. \tag{4.2}$$

The constant values γ_1, γ_2 and γ_3 are the partial derivatives explained in Eq. (B.5):

$$\gamma_1 = \frac{\partial \frac{1}{\dot{s}}}{\partial E_y}\bigg|_{E_{y,ref},E_{\psi,ref},v_{x,ref}}, \gamma_2 = \frac{\partial^2 \frac{1}{\dot{s}}}{\partial E_\psi^2}\bigg|_{E_{y,ref},E_{\psi,ref},v_{x,ref}}, \gamma_3 = \frac{\partial \frac{1}{\dot{s}}}{\partial v_x}\bigg|_{E_{y,ref},E_{\psi,ref},v_{x,ref}} \tag{4.3}$$

Note that the angular deviation E_y is approximated using the second order Taylor Expression. This is done in order to capture the fact that $\cos(E_{\psi,ref})$ is maximum when $E_{\psi,ref} = 0$. The partial derivatives are given below:

$$\gamma_1 = \frac{\partial \frac{1}{\dot{s}}}{\partial E_{y,ref}} = -\frac{\kappa_s}{v_{x,ref}\cos(E_{\psi,ref})} \tag{4.4}$$

$$\frac{\partial \frac{1}{\dot{s}}}{\partial E_{\psi,ref}} = \frac{\left(1 - E_{y,ref}\kappa_s\right)\tan(E_{\psi,ref})\sec(E_{\psi,ref})}{v_{x,ref}} \tag{4.5}$$

$$\gamma_3 = \frac{\partial \frac{1}{\dot{s}}}{\partial v_{x,ref}} = -\frac{\left(1 - E_{y,ref}\kappa_s\right)}{v_{x,ref}^2\cos(E_{\psi,ref})} \tag{4.6}$$

.

As explained, the angular deviation is derived following the second order approximation as

$$\gamma_2 = \frac{\partial^2 \frac{1}{s}}{\partial E_{\psi,ref}^2} = \frac{\left(1 - E_{y,ref}\kappa_s\right)\sec(E_{\psi,ref})\left(\tan^2 E_{\psi,ref} + \sec^2 E_{\psi,ref}\right)}{v_{x,ref}} \tag{4.7}$$

Filling in the mentioned linearization points and considering that $\tan(0) = 0$ and $\sec(0) = 1$, leaves the following constant values:

$$\begin{aligned}\gamma_1 &= -\frac{\kappa_s}{v_{x,ref}}\\ \gamma_2 &= \frac{1}{2}\frac{1 - E_{y,ref}\kappa_s}{v_{x,ref}}\\ \gamma_3 &= -\frac{1 - E_{y,ref}\kappa_s}{2v_{x,ref}^2}\end{aligned} \tag{4.8}$$

The derived equations in (4.8) finally result in the objective function that aims at maximizing the progress along the center-line of the track:

$$J_{race}(E_y, E_\psi, v_x) = \sum_{k=0}^{N-1} \gamma_1 E_y + \gamma_2 E_\psi^2 + \gamma_3 v_x, \tag{4.9}$$

where k is the discrete step length and N the prediction horizon. Eq. (4.9) encourages progress maximization along the center-line s, since the first term causes maximum lateral deviation for curves. In other words, if the road turns left ($\kappa_s > 0$), then

$E_y > 0$ and vice versa causing the vehicle to seek the apex of the curve. The second term minimizes the angular deviation with respect to the center-line of the track, making the choice of Taylor approximating around $E_\psi = 0$ a valid one. Lastly, the third term causes maximization of longitudinal velocity v_x.

4.2.2 Smoothness

During first simulations, the generated trajectories were found to be quite static and not smooth at all. Especially for higher discretization steps Δ_s less precise results were obtained. A smaller step length would yield more precise results, but more iterations are needed to allow planning over the same track segment. Therefore, to enable feasibility and enforce smoothness of the trajectory even with greater step lengths, an additional cost term J_{smooth} is proposed:

$$J_{smooth}(\kappa_v) = \sum_{k=1}^{N-2} \mid \kappa_v[k-1] - 2\kappa_v[k] + \kappa_v[k+1] \mid . \tag{4.10}$$

where k is again the discrete step length and J_{smooth} can be considered a measure of the second derivative of κ_v in discrete form and can be seen as a criterion for comfort. The rationale behind this comes form common practice, where sudden changes in the curvature of the vehicle path highly deteriorates the trajectory smoothness (Bonab and Emadi, 2019). Considering no slip-conditions (as we do with the kinematic bicycle model), the path curvature is related to the vertical tire angle and, inherently, to the steering wheel. In practice, the most comfortable way to approach a corner while driving is to steadily turn the steering wheel. Consequently, the derivative of change in steering angle and vehicle curvature is kept constant. Validation of this comfort control objective is done in Chapter 6.

Conclusively, the final control objective can be seen as a measure of maximizing progress along the center-line of the track while ensuring trajectory smoothness, taken into account all the state variables from Table 3.1, by considering

$$\min J(\zeta) = J_{race} + J_{smooth} \tag{4.11}$$

.

4.2.3 Constraints

As explained in Section 4.1, physical limits and environment conditions are implemented within the MPC formulation as certain constraints, which are explained in this section. Since the dynamics are modeled within the curvilinear coordinate system as derived in Chapter 3, the lane boundaries are defined by spatially dependent state bounds as:

$$E_{y\,\min} \leq E_y \leq E_{y\,\max}. \tag{4.12}$$

Recall from section 3.2 that the spatial reformulation of the dynamics allows natural formulation of both static and dynamic obstacles. In the presence of other vehicles, the bounds on (4.13) can be changed assumed that route information of the other vehicle is known. Depending on this route, a driving environment can be determined based on heuristics (in this case overtaking left or right). Basically, this means that the lane boundaries are narrowed for specific parts of a given race track.

Considering a racing environment, only forward motion of the car is assumed. Expressed in the curvilinear coordinate system, this can be defined by spatially dependent bounds as well:

$$E_{\psi\,\mathrm{min}} \leq E_\psi \leq E_{\psi\,\mathrm{max}}, \tag{4.13}$$

where $E_{\psi\,\mathrm{min}} = (-\frac{\pi}{4}, 0]$ and $E_{\psi\,\mathrm{max}} = [0, \frac{\pi}{4})$.

The curvature rate C_v and longitudinal acceleration a_x as control inputs are limited by constant bounds as:

$$\begin{aligned} v_{x\,\mathrm{min}} &\leq v_x \leq v_{x\,\mathrm{max}}, \\ a_{x\,\mathrm{min}} &\leq a_x \leq a_{x\,\mathrm{max}}, \\ \kappa_{v\,\mathrm{min}} &\leq \kappa_v \leq \kappa_{v\,\mathrm{max}}, \\ C_{v\,\mathrm{min}} &\leq C_v \leq C_{v\,\mathrm{max}}. \end{aligned}$$

Note that all constraints are spatially bounded since the vehicle dynamics are spatially expressed. Formulation in this way results in lateral vehicle dynamics (E_y and E_ψ) as in Eq. (3.11a) and (3.11b) being independent of the longitudinal velocity v_x. This in-dependency enables constant spatial discretization of steps Δ_s, since the discretization is not influenced by vehicle speed in this way.

4.3 Linear Space Varying (LSV)-Model

4.3.1 Linearization

Linear time varying (LTV) models obtained by linearization are employed to construct a tractable convex optimization problem to be solved at each discrete sampling time. This model is successively linearized around a valid equilibrium point at each time step in order to allow posing a convex optimization problem. (Katriniok and Abel, 2011) have shown that linearizations of nonlinear vehicle models over the considered prediction horizon improves the accuracy of the prediction and conqequently the performance of the controller. They compared it to an LTV-MPC controller that considers a static vehicle model (i.e. linearizations remain unchanged over the prediction horizon), which led to remarkable results. A commonly used linearization point in vehicle control is linearizing around the center-line of the track itself, when the goal is to steer the state of the system to the center-line (i.e. $(E_y, E_\psi) = (0, 0)$ as in (Falcone et al., 2008)). As explained earlier, racing lines are quite the opposite of steering the car around the center-line. In most cases, the racing line makes use of the entire width of track to lengthen the radius of a turn as explained in Chapter 2. For this reason, linearizing around the center-line of the track would probably not give a sufficient approximation of the nonlinear function. Alternatively, using solutions of previous iterations from the optimization problem as reference $(\bar{\zeta}, \bar{u})$ for successive linearization gives more appropriate approximations.

$$\begin{pmatrix} E_y' \\ E_\psi' \\ v_x' \\ \kappa_v' \end{pmatrix} = \zeta' = f(\zeta, u) = \begin{pmatrix} f_1(\zeta, u) \\ f_2(\zeta, u) \\ f_3(\zeta, u) \\ f_4(\zeta, u) \end{pmatrix} = \begin{pmatrix} \left(1 - E_y\kappa_s\right)\tan(E_\psi) \\ \frac{\left(1 - E_y\kappa_s\right)\kappa_v}{\cos(E_\psi)} - \psi'^\sigma \\ \frac{\left(1 - E_y\kappa_s\right)}{v\cos(E_\psi)} a_x \\ \frac{\left(1 - E_y\kappa_s\right)}{v\cos(E_\psi)} C_v \end{pmatrix} \tag{4.14}$$

The strategy in which the information gained in solving the previous problem is used in choosing a starting point in an IPM is known as a warm-start strategy (Shahzad, Kerrigan, and Constantinides, 2010). In this case, it is implemented in a way that a proper initial guess acts as warm-start for the MPC optimization problem given in section 4.4.

By linearizing around the solution of previous iterations, the following holds:

$$\dot{\zeta} = \begin{bmatrix} \frac{\partial f_1}{\partial E_y} & \frac{\partial f_1}{\partial E_\psi} & \frac{\partial f_1}{\partial v_x} & \frac{\partial f_1}{\partial \kappa_v} \\ \frac{\partial f_2}{\partial E_y} & \frac{\partial f_2}{\partial E_\psi} & \frac{\partial f_2}{\partial v_x} & \frac{\partial f_2}{\partial \kappa_v} \\ \frac{\partial f_3}{\partial E_y} & \frac{\partial f_3}{\partial E_\psi} & \frac{\partial f_3}{\partial v_x} & \frac{\partial f_3}{\partial \kappa_v} \\ \frac{\partial f_4}{\partial E_y} & \frac{\partial f_4}{\partial E_\psi} & \frac{\partial f_4}{\partial v_x} & \frac{\partial f_4}{\partial \kappa_v} \end{bmatrix} (\zeta - \bar{\zeta}) + \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix} (u - \bar{u}) + f(\bar{\zeta}, \bar{u}) \tag{4.15}$$

Since the vehicle dynamics are expressed in spatial coordinates, the model will be referred to as Linear Space Varying (LSV)-Model.

4.3.2 Discretization

Now that a linearized vehicle model is obtained, discrete matrices A_d and B_d are found using Zero Order Hold (ZOH) discretization as extensively explained in Appendix B. In the case of matrix A_c being nonsingular (i.e. invertible), the discretizated matrices can be approximated as

$$M(k) = \exp\left(\begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix} \Delta_s\right) = \begin{bmatrix} A_d^{\text{ZOH}} & B_d^{\text{ZOH}} \\ 0 & I \end{bmatrix} \tag{4.16}$$

The optimization problem is constrained by the space-varying vehicle dynamics as:

$$\zeta(k+1) = A_d^{\text{ZOH}}(k)\zeta(k) + B_d^{\text{ZOH}}(k)u(k) + H_c^{\text{ZOH}}(k), k \geqslant 0 \tag{4.17}$$

Following the steps in appendix B leads the the following matrices describing these final dynamics:

$$A_d^{\mathrm{ZOH}}(k) = \begin{bmatrix} \kappa_s \tan \bar{E}_\psi & -\frac{(1-\kappa_s)}{\cos^2 \bar{E}_\psi} & 0 & 0 \\ -\frac{\kappa_s \bar{\kappa}_v}{\cos \bar{E}\psi} & \frac{\left(1-\bar{E}_y\kappa_s\right)\bar{\kappa_v} \tan \bar{E}_\psi}{\cos \bar{E}_\psi} & 0 & \frac{\left(1-\bar{E}_y\kappa_s\right)\bar{\kappa}}{\cos \bar{E}_\psi} \\ \frac{\kappa_s(\mu\bar{v}_x-\bar{a})}{\bar{v}_x \cos \bar{E}_\psi} & \frac{\left(\bar{E}_y\kappa_s-1\right)\left(\mu\bar{v}_x-\bar{a}\right) \tan \bar{E}_\psi}{\bar{v}_x \cos \bar{E}_\psi} & -\frac{\left(1-\bar{E}_y\kappa_s\right)\bar{a}}{\bar{v}^2 \cos \bar{E}_\psi} & 0 \\ -\frac{\kappa_s \bar{C}}{\bar{v}_x \cos \bar{e}_\psi} & \frac{\left(1-\bar{E}_y\kappa_S\right)\bar{C} \tan \bar{E}_\psi}{\bar{v}_x \cos \bar{E}_\psi} & -\frac{\left(1-\bar{E}_y\kappa_s\right)\bar{C}}{\bar{v}_x^2 \cos \bar{E}_\psi} & 0 \end{bmatrix}, \tag{4.18}$$

$$B_d^{\mathrm{ZOH}}(k) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{\left(1-\bar{E}_y\kappa_s\right)}{\cos \bar{E}_\psi} & 0 \\ 0 & \frac{\left(1-\bar{E}_y\kappa_s\right)}{\cos \bar{E}_\psi} \end{bmatrix}. \tag{4.19}$$

The spatial state vector ζ and control vector u are given in table 3.1 and table 3.2.

4.4 Final Control Structure

Based on the previous sections, the final structure of the model predictive control algorithm will be derived. Besides the vehicle dynamics of Section 3.2 and the overtaking constraints discussed in Section 3.2, some assumptions have to be made while designing the algorithm. These assumptions are summarized below.

Assumption 1: *The subject vehicle drives at maximum possible velocity and, therefore, overtakes the other vehicle at maximum possible velocity.*

Assumption 2: *The vehicles only have forward motion, which can be justified by the vehicles being in a racing environment.*

Assumption 3: *The center line of the track is used as reference line for both vehicles and information about the track parameters is known.*

Assumption 4: *Drivers enter the track while racing and state measurements at this time is available, making the warm-start strategy with appropriate initial guesses a valid choice.*

At each discretization step length Δ_s, the spatial-based MPC controller solves the following constrained convex optimization problem:

$$
\begin{aligned}
\underset{\bar{u},\alpha}{\text{minimize}} \quad & J_{race}(E_y, E_\psi, v_x) + J_{smooth}(\kappa_v) + \eta\alpha^T\alpha && \text{(4.20a)}\\
\text{subject to} \quad & \zeta_{k+1} = A_{d,k}^{\text{ZOH}}\zeta_k + B_{d,k}^{\text{ZOH}}u_k + H_{d,k}^{\text{ZOH}}, && \text{(4.20b)}\\
& \zeta_0 = \zeta(s), && \text{(4.20c)}\\
& E_{y\,\text{min}} - \alpha_{E_y} \leq E_y \leq E_{y\,\text{max}} + \alpha_{E_y}, && \text{(4.20d)}\\
& E_{\psi min} - \alpha_{E_\psi} \leq E_\psi \leq E_{\psi\,\text{max}} + \alpha_{E_\psi}, && \text{(4.20e)}\\
& v_{x\,\text{min}} - \alpha_{v_x} \leq v_x \leq v_{x\,\text{max}} + \alpha_{v_x}, && \text{(4.20f)}\\
& \kappa_{v\,\text{min}} - \alpha_{\kappa_v} \leq \kappa_v \leq \kappa_{v\,\text{max}} + \alpha_{\kappa_v}, && \text{(4.20g)}\\
& a_{x\,\text{min}} \leq a_x \leq a_{x\,\text{max}}, && \text{(4.20h)}\\
& C_{v\,\text{min}} \leq C_v \leq C_{v\,\text{max}}, && \text{(4.20i)}\\
& \alpha_{E_\psi} \geq 0, \alpha_{E_y} \geq 0, \alpha_{v_x} \geq 0, \alpha_{\alpha_\kappa} \geq 0, && \text{(4.20j)}\\
& k = 0, \ldots, N-1 && \text{(4.20k)}
\end{aligned}
$$

The objective function (4.20a) is described in section 4.2. Additionally, a slack variable vector α is implemented in order to increase the feasibility region of the constraints. By doing so, initial hard constraints are softened and the optimal solution is easier to find. The penalization diagonal $\eta = diag(\eta_{E_y}, \eta_{E_\psi}, \eta_{v_x}, \eta_{\kappa_v})$ is added to push the slack variables toward zero, causing the problem to stay as much as possible within the initial hard constraints. It is assumed that these possible minor overshoots of the bounds is valid. For example, in the case of bounds on E_y, the race car is preferred to generate a trajectory slightly outside the track bounds over immediate shutdown (i.e. when no solution is found). The vehicle dynamics from constraint (4.20b) are derived in section 4.3 and the initial state ζ_0 is estimated by proper guess (i.e. the warm start) $\zeta(s)$ in constraint (4.20c). The spatial states and control inputs are limited by constant bounds as explained in section 4.2.3 and can be found in the MPC scheme as constraints (4.20d) – (4.20k).

Chapter 3 and 4 describe the proposed vehicle model and the derivation of the MPC scheme. Conclusively, the complete working mechanism of the spatial-based MPC algorithm is summarized by means of a pseudo-code on the next page. Recall that discretization is done in terms of spatial parameter Δ_s. This means that the maximum number of iterations i_{max} to complete a specific track segment depends on the length of that segment.

Algorithm 1: Pseudo-code Spatial-Based MPC

```
1: Initialize warm start $\zeta_0 = \zeta(s)$ to compute optimal input sequence $\bar{u}$
2: for $i$ = 0, . . . , $i_{max}$ do
    3: for $k$ = 0, . . ., N-1 do
        Linearize around $(\bar{\zeta}_k, \bar{u}_k)$ (4.15) and discretize (4.16);
        Solve MPC formulation (4.20a)-(4.20k) with CVXPY;
        if Solution is optimal then
            $\bar{\zeta}_k = \zeta_k$ , $\bar{u}_k = u_k$ ;
        else
            Solver cannot solve MPC;
        end
    end
    Recover global coordinates (3.15)
end
```

Chapter 5

Simulation Setup

5.1 Nonholonomic vs. holonomic robots

The omniwheel robots often used in the DTPA lab are unique in the sense that they are able to move freely in two directions, enabling the conversion from a non-holonomic (car-like models) to a holonomic robot. Non-holonomic robots, considered in this thesis, have two controllable degrees-of-freedom. In other words, forward/backward motion and turning is possible.

Perfect tracking of certain reference trajectories is achievable only if the reference trajectory is suitable for the physical vehicle. Meaning that a trajectory, which is feasible for a unicycle-type vehicle, is not necessarily feasible for a car-like vehicle (such as modelled with the kinematic bicycle model) (Morin and Samson, 2008). For this reason, performing simulations on the Nexus robots represented by mathematical expressions derived from the bicycle model would not be representative for real world situations. Therefore, the nexus robots will be outside the scope of this thesis.

5.2 Simulation Environment

Since the used kinematic bicycle model does not match the omniwheel robots used in the DTPA lab of the University of Groningen (RUG), these robots cannot be used for simulation. Therefore, commonly used simulation software like Gazebo will not be used in this project. However, since Gazebo is compatible and works well with software like Robot Operating System (ROS) in combination with Python, the latter one is used for coding and simulation. All results are generated in Python with the module called *Matplotlib*. Since simulation in Gazebo with a representative simulation robot (an Ackermann Steering Robot for example) could be a future project, Python as programming language is useful due to its compatibility with ROS. The optimization problem as in (4.20a) - (4.20k) is modeled as an convex optimization problem in a Python-embedded modeling language called *CVXPY* (Diamond and Boyd, 2016). Within The problem is initialized with proper guesses and subsequently, the convex optimization problem is solved at each sampling time k, over the prediction horizon N with step length Δ_s.

As explained in Chapter 4, the racetrack is manually created and imported into the simulation environment by using a cubic-spline planner where the spline is representing the center-line of the track. Boundary conditions such as road width are generated in the same fashion, resulting in track visualization according to Figure 5.2.

5.3 F1 Suzuka Circuit Japan

As explained in Section 5.2, the center-line of the track is manually created and imported into the simulation environment. To simulate the performance of the controller design, the Formula 1 Suzuka Circuit (Figure 5.1) is considered and replicas of different segments of the track are manually created with the cubic spline planner and imported into the simulation environment.

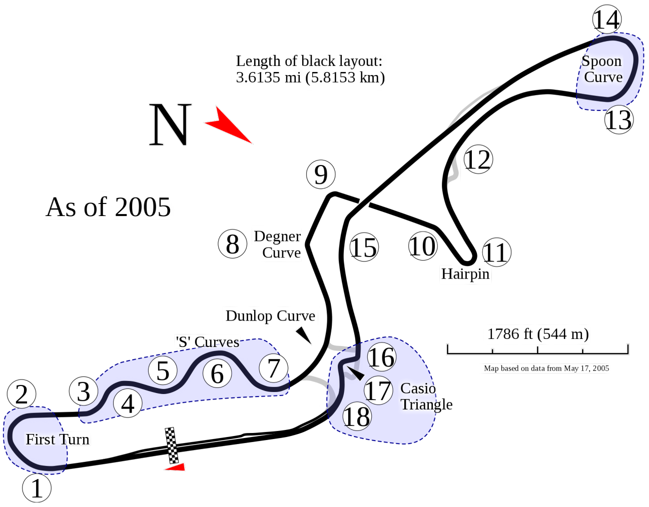
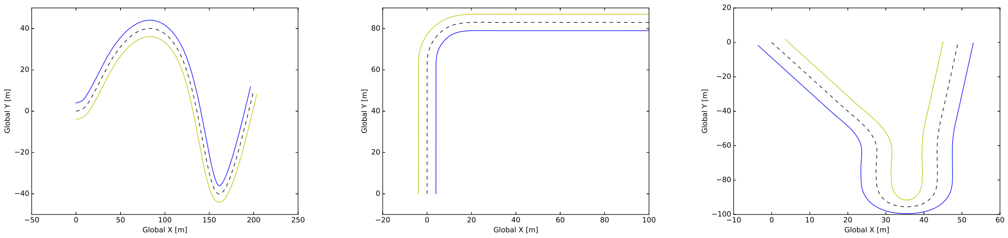


FIGURE 5.1: F1 Suzuka Circuit, Japan

From Figure 5.1, an approximation of respectively the S-curve (3-4-5), the 90-degrees turn (9) and U-turn of the hairpin (10-11) are created with the cubic spline planner and used for simulation purposes. The result of running these segments through the cubic spline planner are given in the figure below.



(A) The S-curve

(B) The 90-degrees curve

(C) The hairpin curve

FIGURE 5.2: Representation of Track Segments

Chapter 6

Simulation and Results

This chapter is devoted to the obtained results from the simulations. Section 6.1 is meant to validate whether the cost function as proposed in Section 4.2 is performing as scheduled or not. Subsequently, the effect of varying friction coefficient μ on the generated trajectory is analyzed in Section 6.2. Finally, Section 6.3 tests the robustness of the controller in terms of varying planning horizon and step length, track variation, initial conditions and in the presence of race opponents. The values with which simulations are done are given in each section. However, constraints as in (4.20d) – (4.20j) stay within the following bounds:

$$\begin{aligned}
4 - \alpha_{E_y} \leq E_y &\leq 4 + \alpha_{E_y} \\
-\frac{\pi}{4} - \alpha_{E_\psi} \leq E_\psi &\leq \frac{\pi}{4} + \alpha_{E_\psi} \\
0 - \alpha_{v_x} \leq v_x &\leq \frac{150}{3.6} + \alpha_{v_x} \\
-\frac{\tan(\delta_{\max})}{L} - \alpha_{\kappa_v} \leq \kappa_v &\leq \frac{\tan(\delta_{\max})}{L} + \alpha_{\kappa_v} \\
-5 \leq a_x &\leq 5 \\
-0.2 \leq C_v &\leq 0.2
\end{aligned}$$

With maximum steering angle $\delta_{\max} = \frac{\pi}{4}$ and length of the car $L =$ m. In all simulations, the slack variables α are present in the objective function and pushed towards zero as much as possible.

6.1 Validation

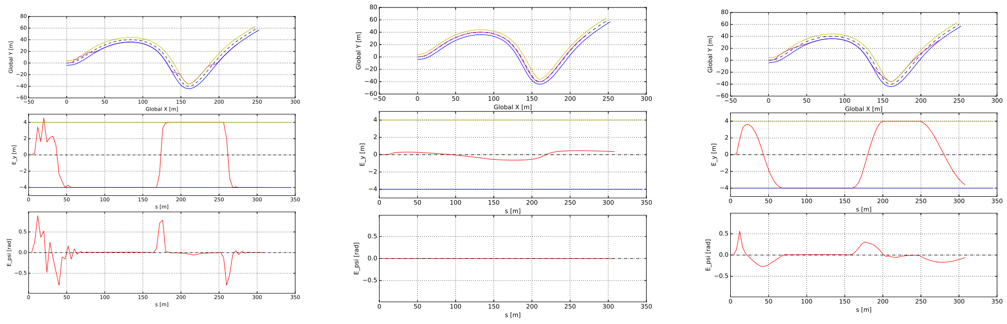
6.1.1 Validation of J_{race}

In order to validate whether the LSV-MPC scheme is functioning according to the formulated problem, each segment of the cost function is taken into consideration separately. Different experiments are executed to validate the functionality of the cost function in terms of solve time and solve cost. The initial values for the validation are given in Table 6.1. These initial values indicate that the vehicle enters the track in a straight way at the center-line with 40 m/s, showing no angular deviation.

State/parameter	Unit	Value
E_y	m	0
E_ψ	rad	0
v_x	m/s	40
κ_v	m^{-1}	0
N	-	10
Δ_s	m	4
μ	-	0

TABLE 6.1: Initial values for validation

The results of solving the MPC scheme by simulating the cost segments separately are given in Figure 6.1. Recall the rationale behind the cost function, as it tries to maximize the progress along the center-line of the track. The first term γ_1 tries to maximize the lateral deviation along with the curve of the track. It can clearly be seen that once the prediction horizon recognizes a curve is coming up, the controller steers the vehicle to the boundary of the track. Figure 6.1a validates that this objective segment is reached. The functionality of γ_2 in the objective is to minimize the heading angle with respect to the center-line of the track. In practice this means it will try to follow the track along the center-line, as shown in Figure 6.1b When only γ_3 was considered as objective, the solver could not solve the MPC scheme. Therefore, Figure 6.1c shows the three components all together. Additional experiments were conducted by making combinations of the three segments. It was experienced that adding γ_3 to the combination of γ_1 and γ_2 did not change the generated trajectory. This coincides with statements made in Chapter 4, where the discretization was said to be independent of longitudinal velocity v_x. This value increased linearly to its maximum $v_{\max}$, but did not affect the lateral dynamics (E_y, E_ψ). Further analysis on this subject is done in Section 6.2.



(A) First term: γ_1

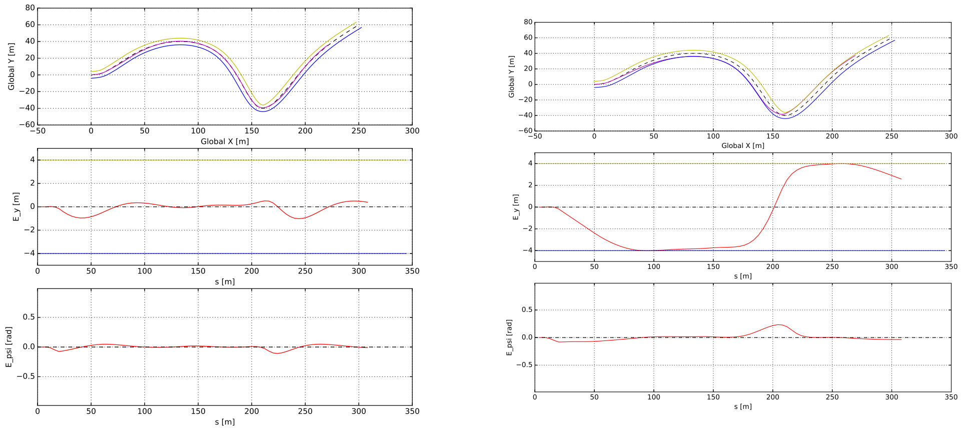
(B) Second term: γ_2

(C) Complete term J_{race}

FIGURE 6.1: Validation of objective function J_{race}

6.1.2 Validation of J_{smooth}

To show the effect of J_{smooth} on the predicted trajectory, first the term is analyzed separately (i.e. cost objective only dependent on J_{smooth}). Subsequently, J_{race} is added to see the effect of simultaneously optimizing both J_{race} and J_{smooth}.

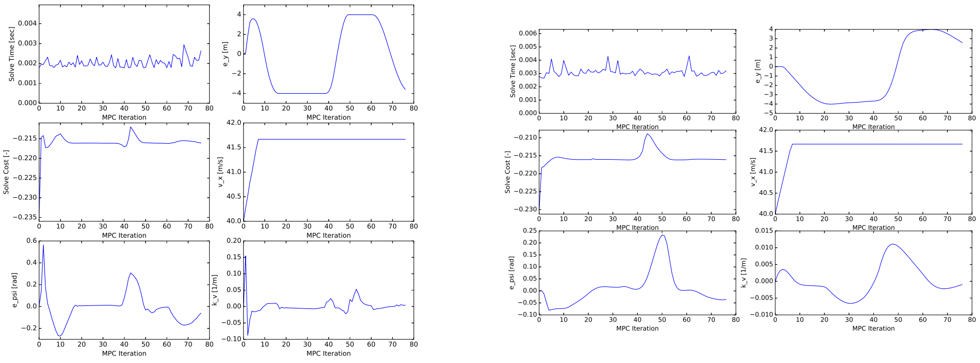


(A) J_{smooth} considered separately

(B) $J_{race} + J_{smooth}$

FIGURE 6.2: Validation of J_{smooth}

Figure 6.2a shows the effect when only J_{smooth} is considered as objective. It can be clearly seen that lateral- and angular deviations from the center-line are small and it aims at minimizing the vehicle curvature only. Figure 6.2b shows the effect of including both components J_{race} and J_{smooth}, generating a trajectory that maximized the progress along the center-line of the track, while ensuring curvature smoothness at the same time. Additionally, the spatial states and some solver information is given in Figure 6.3 for both J_{race} separate and J_{race} and J_{smooth} combined.



(A) J_{race} considered separately

(B) $J_{race} + J_{smooth}$

FIGURE 6.3: MPC iterations vs. spatial states

An aspect that an addition in the objective function could endanger is the computation time. To check whether the addition of J_{smooth} could still be feasible for real-time application in terms of efficient computation times, the solving times per iteration are summarized in Figure 6.3 and it can be seen that the effect of adding J_{smooth} only has a minor influence in terms of solving time per MPC iteration.

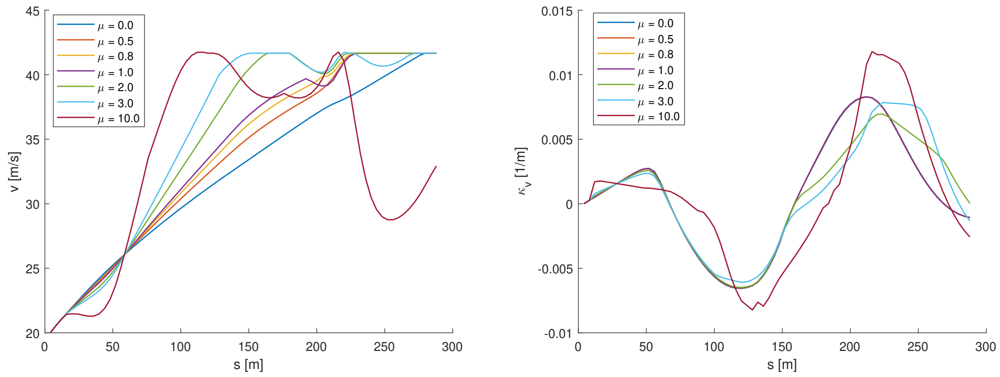
6.2 Friction Coefficient μ

Recall the recurring statements that the lateral dynamics (E_y, E_ψ) from Eq. (3.11a) and (3.11b) are independent of the longitudinal speed. This section is meant to show the effect of changing friction parameter μ on the generated trajectory. Despite the fact that maximum longitudinal speed is assumed most of the time, these simulations were done by setting the initial longitudinal speed to 20 m/s in order to investigate the speed profile on the S-curve and Hairpin curve. These simulations ran with horizon length $N = 15$ and discretization step $\Delta_s = 4$.

6.2.1 μ on S-curve

Figure 6.4 summarizes the effect of enlarging the value for μ for the spatial states v_x and κ_v on the S-curve. Besides evolution of these two states, the results of lateral dynamics are given in Appendix D and predefined track information in Appendix C.2. From Figure 6.4a is visible that μ is modeled within the dynamics such that an increase in μ leads to a higher increase velocity (i.e. higher acceleration). Setting μ to zero gives an approximately linear increase in velocity, until it hits the maximum value $v_{x\,\max}$, which was already stated in Section 6.1.1. The change in curvatures of the velocity profiles can be connected to the curves present on the track segment. As example, take $\mu = 10$ in Figure 6.4a and 6.4b and compare it to track information given in C.1a. The lower μ-values do not affect the velocity- and car curvature profiles in the first turn of the S-curve. However, in the second (and sharper) turn of the S-curve, both velocity and car curvature display odd behaviour.

When comparing the μ-values to the generated trajectories (Appendix D.1), it appears that the controller fails to steer the vehicle to the apex of the second curve. More precise, it captures the fact that when the vehicle rapidly accelerates, its unable to make sharp turns to some extent. It can be concluded that the higher μ gets, the faster the vehicle decelerates on higher curvatures of the track and the faster it accelerates on straight segments. With these observations, μ can in fact be seen as a friction component for the vehicle dynamics. Neglecting μ causes the controller to request acceleration at all time. By including μ, the controller is able to capture the dynamics more realistically as it changes the velocity profile in curvy segments of the track.



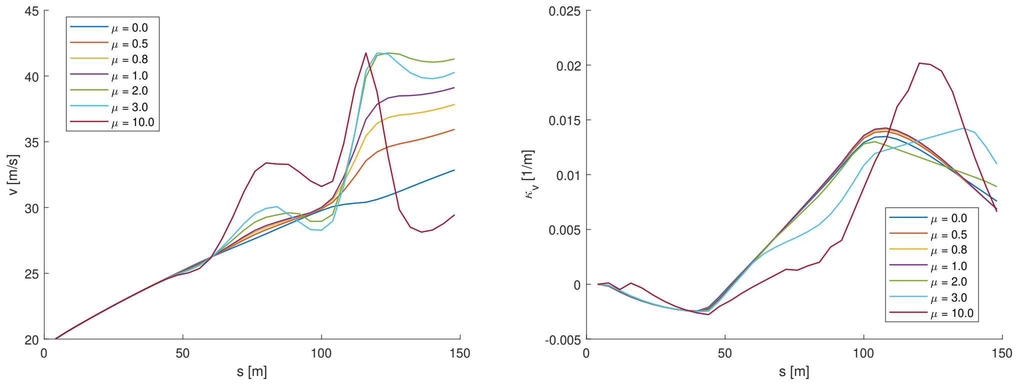
(A) Velocity profile

(B) Vehicle curvature profile

FIGURE 6.4: Variation of friction values μ on S-curve

6.2.2 μ on Hairpin Curve

The same simulations were done for the hairpin curve and yields the same observations, as can be seen in Figure 6.5. When μ obtains a value larger than 1, it starts to affect the velocity profile and car curvature profile as in Figure 6.5a and 6.5b. Consequently, the optimized trajectories (Appendix D.2) start to change for these values as well.



(A) Velocity profile

(B) Vehicle curvature profile

FIGURE 6.5: Variation of friction values μ on Hairpin curve

Since it was noted that $\mu > 1$ shows unfavourable behaviour in the evolution of the spatial states, further simulations are done by setting $\mu = 0.8$.

6.3 Robustness Analysis

Theoretically, the proposed controller should be robust with respect to uncertainties and disturbances due to the successive linearization and the Receding-Horizon strategy. This section is meant to show the robustness of the proposed controller in terms of varieties in conditions/parameters. It is of high importance that robustness is verified, otherwise certain statements on the performance are not valuable. For example, if only one track segment is considered during simulation, nothing can be concluded on performance in general, only on the performance of the design on that particular segment. This section is meant to analyze the robustness of the MPC controller with regard to different tracks, a variation in parameters and presence of race opponents.

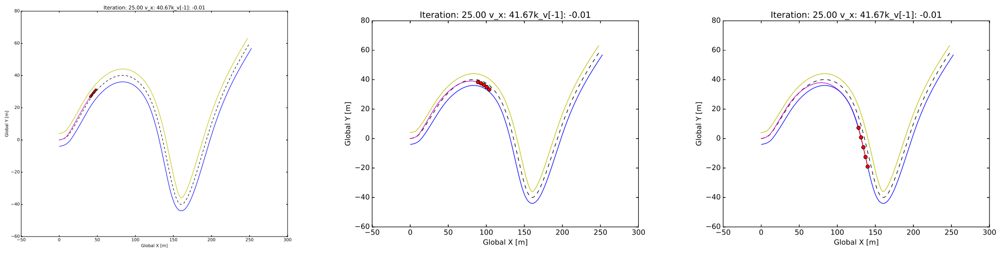
6.3.1 Prediction Horizon

The change change in step length Δ_s and planning horizon N matters a lot in this thesis, since it determines how far up ahead the track the predictions are done. Due to methods elaborated on in Chapter 4, a relatively big discretizion step is allowed. This subsection is meant to illustrate the effect of varying horizon (N) and step (Δ_s) lengths on the simulation. For this experiment, the S-curve as in Figure 5.2a is used with varying horizon and step length compositions as shown in Table 6.2. For these different compositions of N and Δ_s, the number of iterations necessary to complete the track segments, the sum of the cost function and the total solve time are computed.

N	5			10		
$\Delta_s[m]$	**2**	**4**	**6**	**2**	**4**	**6**
# Iterations	167	81	52	162	76	47
$\sum J(\zeta)$	-16.2	-7.9	-5.2	-35.3	-16.6	-10.4
$\sum t_{solve}[sec]$	0.19	0.10	0.06	0.51	0.24	0.15
N	**15**			**20**		
$\Delta_s[m]$	**2**	**4**	**6**	**2**	**4**	**6**
# Iterations	157	71	42	152	66	37
$\sum J(\zeta)$	-53.3	-24.1	-14.5	-69.8	-30.0	-17.1
$\sum t_{solve}[sec]$	0.98	0.40	0.25	1.43	0.66	0.39

TABLE 6.2: Effect of varying planning horizon and step length

It can be seen from Table 6.2 that higher horizon lengths N result in a more time-consuming optimization problem, which corresponds to statements made in Section 4.1. Choosing a composition of N and Δ_s is a matter of trade-off between computation-time and accuracy of the results. A longer horizon length N and a greater step length Δ_s both enable to plan further on the track. However, increasing N lead to an increase in computation time and enlarging Δ_s yields less precise results. As example, Figure E.2 shows the effect of taking a constant horizon length $N = 5$ and varying step length $\Delta s = [2, 4, 6]$ at the same iteration number. The $N - 1$ red dots (as in Eq. (4.20a)) indicate the predictions (around which the vehicle dynamics are linearized in the next iteration). The purple line is the trajectory the vehicle has driven so far. It can be clearly seen that the algorithm is able to plan further ahead with a greater step length Δ_s. Notice also that in Figure 6.6b, the vehicle is turning to the inner bound of the track much later than in Figure 6.6c, as it recognizes the turn later as well. Appendix E.1 contains the corresponding trajectories of Figure E.2 over the whole track segment.



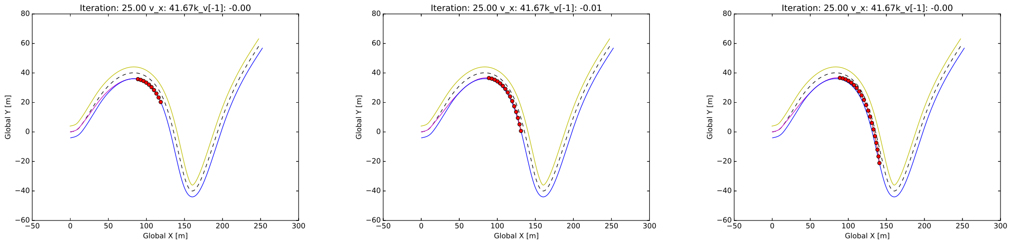
(A) $N = 5, \Delta_s = 2$

(B) $N = 5, \Delta_s = 4$

(C) $N = 5, \Delta_s = 6$

FIGURE 6.6: Constant $N = 5$ and changing Δ_s

When simulating it the other way around by keeping Δ_s constant and varying with horizon length N, similar results are depicted in Figure 6.7. The vehicle has not traveled further along the track as N becomes larger, but is does plan further ahead. Notice that the larger N, the sooner the vehicle recognizes the turn is coming up. Consequently, it drives to the inner bound of the track earlier. Especially in Figure 6.7a and 6.7b, this difference is visible. Appendix E.2 contains the corresponding trajectories of Figure 6.7 over the whole track segment. Looking at Figures E.2b and E.2c, the trajectory of the first curve is identical, since for both $N = 15$ and $N = 20$ the prediction horizon takes the curve into account from the start. In the second curve, however, a horizon length of 20 identifies the curve sooner and steers to the apex almost instantly after the first curve.



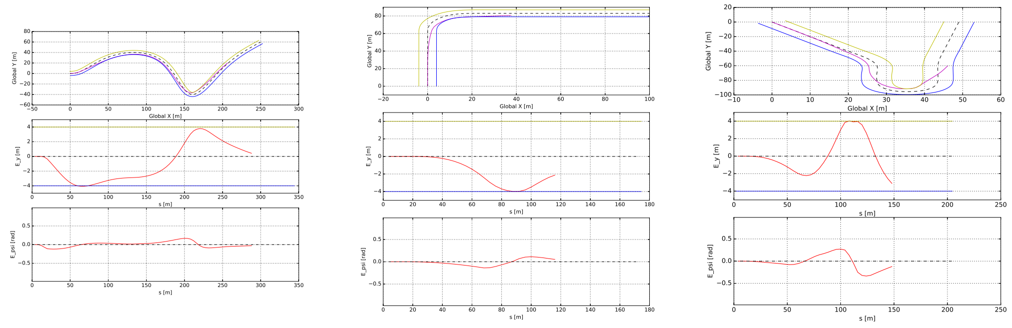
(A) $N = 10, \Delta_s = 4$ (B) $N = 15, \Delta_s = 4$ (C) $N = 20, \Delta_s = 4$

FIGURE 6.7: Constant $\Delta_s = 4$ and changing N

After the simulations and looking at Table 6.2, it can be concluded that the MPC controller is robust with respect to a change in planning horizon N and step length Δ_s. For the upcoming simulations, $N = 15$ and $\Delta_s = 4$ will remain constant.

6.3.2 Track Variation

Considering the whole track, different segments of the track are taken to simulate the overall performance of the proposed controller. This subsection shows the MPC controller performance on the S-curve, the 90-degrees turn and the hairpin curve, respectively given in Figures 5.2a, 5.2b and 5.2c. The results of simulating the vehicle on these track segments are summarized in Figure 6.8.



(A) Results S-curve

(B) Results 90-degrees turn (C) Results hairpin

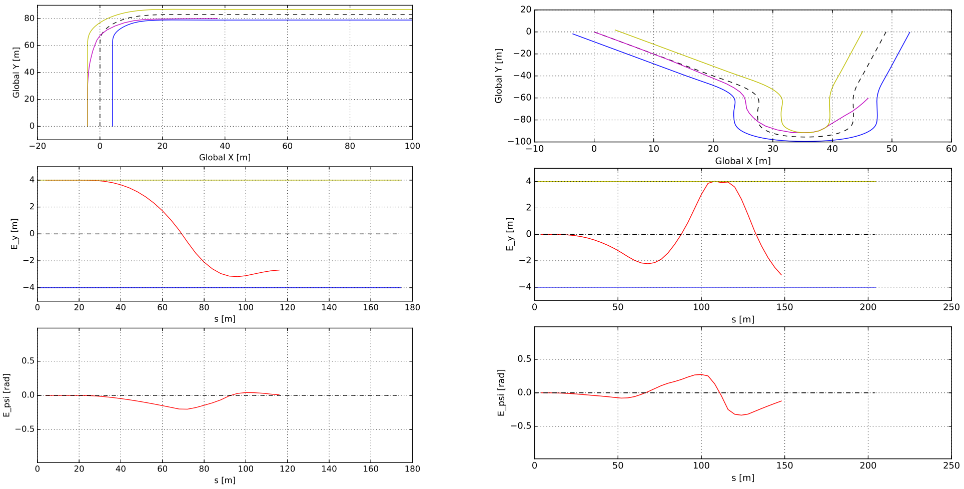
FIGURE 6.8: Simulation on different track segments

As observed in Figure 6.8 and in other sections, the MPC controller is able to generate satisfactory results on different tracks with predefined track information as in Appendix C. Figure 6.8 is considered sufficient to conclude satisfactory performance of the MPC controller.

6.3.3 Initial Conditions

A variation in initial conditions for the problem formulated in this thesis, concerns the initial position, heading, velocity and car curvature of the simulated vehicle. Since properly guessed initial conditions are necessary to get satisfying results, varying with these initial condition is a good way to show the performance of the controller. The change in initial conditions is tested on the 90-degrees turn and the hairpin, as in Figures 5.2b and 5.2c. The initial lateral displacement is changed in the

90-degrees turn and adjustments for initial velocity are tested on the hairpin curve and corresponding results are depicted in Figure 6.9.



(A) $E_{y,0} = -4$ on 90-degrees turn

(B) $v_{x,0} = 20$ on hairpin

FIGURE 6.9: Variation in initial values

Considering Figure 6.9a, the vehicle steers toward the curve as soon as it recognizes it. This behaviour is similar to results shown in previous sections. In the case of Figure 6.9b, velocity profiles are extensively analysed in Section 6.2. This section complements robustness claims of the MPC controller with respect to changing initial conditions.

6.3.4 Overtaking Manoeuvres

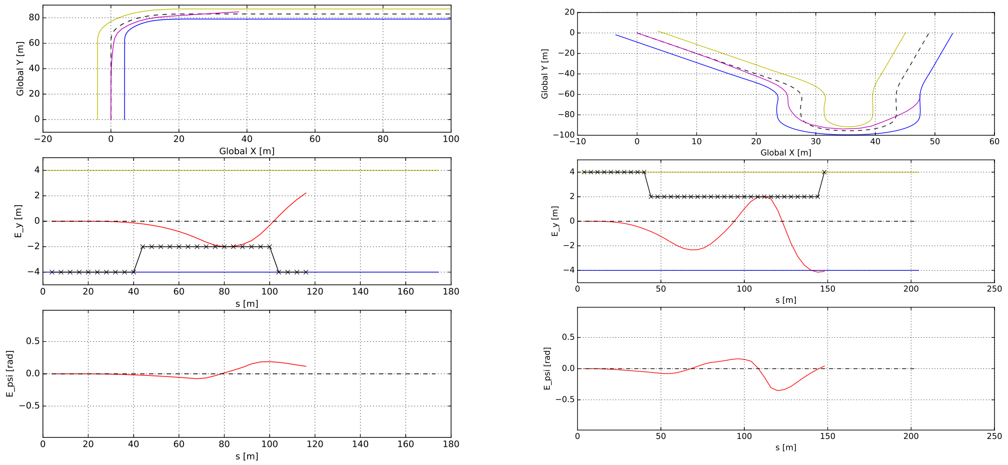
In general, there are two ways to formulate an obstacle within the problem. Firstly, obstacles can be represented by a change in constraints on the driving environment. In this case this means changing the conditions on the inner- and outer bounds of the track. Since the problem is formulated in a way that the simulated vehicle stays within bounds of the track, a change in formulation of the driving corridor can be used to implement the presence of either moving or static obstacles on the track. This method is used often in literature because of its simplicity in terms of adding it to the problem formulation (due to being formulated as linear constraints). Secondly, additional states for overtaking as in (Buyval et al., 2017) can be formulated by implementing dynamics of the obstacle in the problem formulation. However, since the problem is formulated as a convex optimization problem and overtaking dynamics are not particularly convex, adding it as state for the problem at hand is rather difficult and quite possibly inaccurate.

This section shows the performance of the controller in the presence of obstacles. Since the vehicle dynamics are spatially formulated, obstacles can be formulated by means of bounds on the track as well. The overtaking manoeuvre is then heuristically determined, assuming that the trajectory of competitors in the track are known.

In contrary to the bounds on the race track, obstacle bounds should be ideally formulated as hard constraints. As slightly driving outside the race track is not catastrophic, impact on a static or dynamic obstacle is. Consequently, the spatial bounds representing race opponent on the track are modeled without any slack variable α.

Scenario 1: one race opponent on the track segment

This scenario sketches results when one race opponent is present on the track segment, formulated as hard constraint and, consequently, the feasibility region of the MPC problem decreases. Figure 6.10 shows the performance of the controller on the 90-degrees turn and the Hairpin. Since the controller aims at maximizing the progress along the center-line by (among other things) deviating to the inner bound of the track at curves, race competitors are modeled to be at that inner curve. In Figure 6.10a, the opponent drives 2 meters from the inner bound of track in an area of [40m - 100m]. This area is determined based on an arbitrary speed profile of the race opponent. When considering the hairpin curve, the scenario is sketched as in Figure 6.10b.



(A) Scenario 1.1

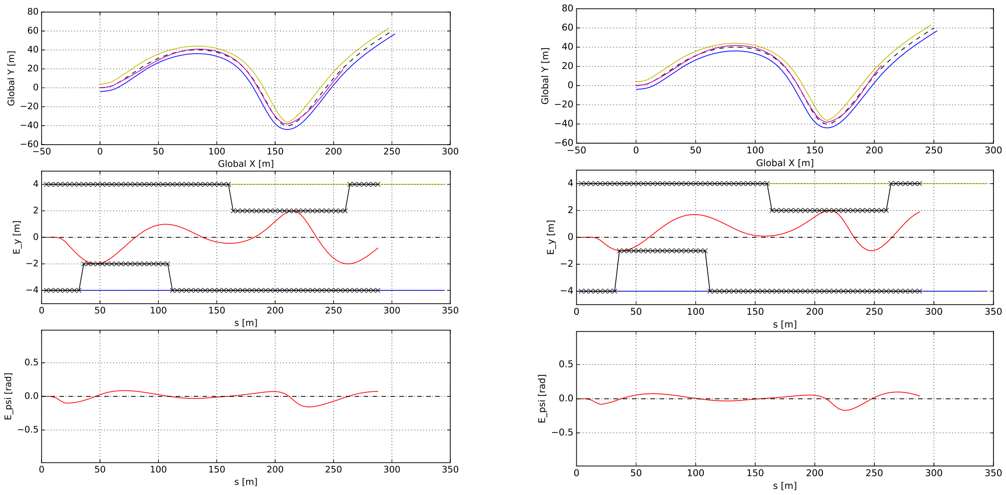
(B) Scenario 1.2

FIGURE 6.10: Scenario 1: one race opponent

The opponents on the track segments are depicted as bounds on the track, indicated by black markers. It can be clearly seen that on both track segments, Figures 6.10a and 6.10b, the car seeks to find a path closest to the other vehicle in the turn. This corresponds to the initial objective of the controller: maximizing progress of the subject vehicle along the center-line. It can also be seen that the vehicle seeks to minimize its angular deviation E_ψ from the center-line immediately after it exits the turn.

Scenario 2: two race opponents on the track

This is were the spatial dynamics show its property of naturally formulation of obstacles. In this case, the race opponent are modeled as a dynamic obstacles driving either 2 or 3 meters from the inner bound of the track. Figure 6.11a shows the scenario where the first car is driving 2 meters from the track in the area of [38m - 108m],



(A) Scenario 2.1

(B) Scenario 2.2

FIGURE 6.11: Scenario 2: two race opponents

in terms of the spatial coordinate system. The second car drives on the inside of the second curve, 2 meters away from the bound. This is modeled as an area of [160m - 240m] in the spatial coordinate system. Figure 6.11b shows a similar scenario where the first race opponent drives 3 meters away from the track bound.

Scenario 3: three race opponents on the track

To check whether the vehicle is able to move its way through two other vehicles, a scenario with three race opponents is considered. It is similar to the scenario sketched in Figure 6.11a, but in this case a second opponent driving parallel to the other opponent in the first curve is considered. It can be observed that the vehicle is able to steer through two other opponents smoothly mainly by minimizing its angular deviation with respect to the center-line of the track. However, the third race opponent is overtaken with quite some lateral distance in between. This can be explained due to the fact that the MPC controller aims to ensure smoothness at all times. By manoeuvring its way through the first two opponents, it does not have enough time to steer to the apex of the second curve and, consequently, chooses to smoothly steer around the third opponent.

Additionally, the simulations in presence of race opponents still show efficient computation times, as illustrated in Figure 6.13. It can be seen that the more race opponents on the track, the higher the mean solve times per iteration. This makes sense, since more race opponents on track means more constraints leading to a more computational heavy problem to solve.

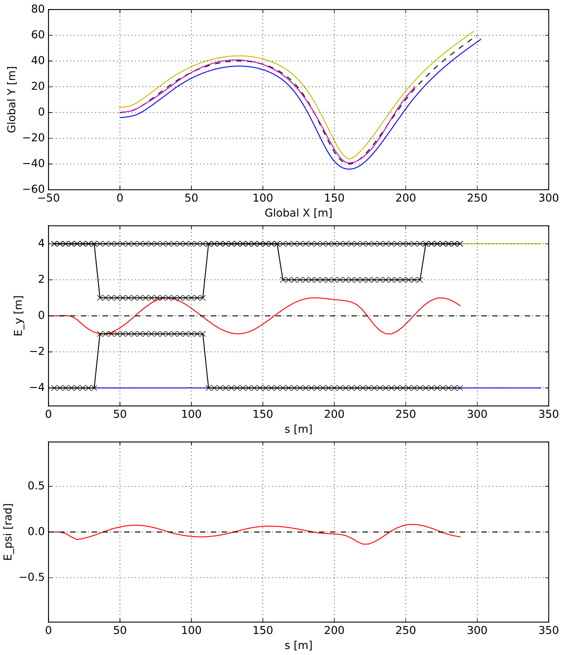
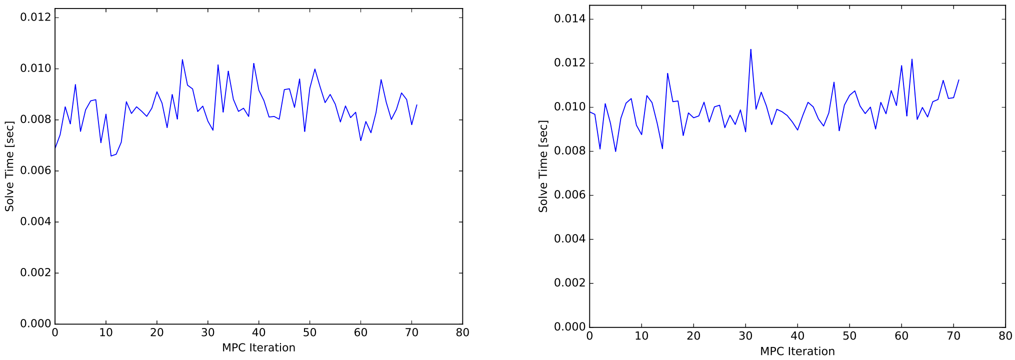


FIGURE 6.12: Scenario 3: three race opponents



(A) Solve time with 2 race opponents

(B) Solve time with 3 race opponents

FIGURE 6.13: Computation times in the presence of race opponents

Chapter 7

Discussion

A big assumption made during the simulation phase is the no-slip condition of the kinematic vehicle model. Assuming perfect road-tire grip at all times is something that in real life cannot be justified, especially not when operating at the limit of friction as in race environments. Since the relative simple kinematic bicycle model is far from an identical representation of the vehicle dynamics of a formula 1 car, all results must be reflected to as such. Theoretically, kinematic models will not be able to capture important non-linearities that arise from tire-road interaction, which defines car behaviour. Such conditions require a highly detailed dynamic model of the car in order to plan the trajectory and to ensure its feasibility. For the problem at hand, these non-linearities are approximated by successive linearization of the vehicle dynamics arising from the kinematic bicycle model.

Another consequence of utilizing a bicycle model instead of a full vehicle model is the neglect of vehicle width. Since the 4-wheel vehicle model is simplified to a single-axis bicycle model, dimension in terms of width is not included. Only the length of the vehicle is included in the vehicle dynamics as constraint in the optimization problem of the maximum curvature of the vehicle. Fortunately, by defining spatial bounds this can be easily overcome by adjusting the bound according to the width of race opponents.

Lateral acceleration is not taken into account due to the simplicity of the vehicle model. Initially, the controller requested constant acceleration on the track segments, independent on the local curvatures of the track. This was the result of the segment of the cost function that aims at maximizing the longitudinal velocity v_x. This effect is attempted to overcome by introducing friction coefficient μ. Enlarging this value showed deceleration before curves and higher acceleration after curves. It was observed that this was related to the curvature of the vehicle. Considering the spatial vehicle dynamics as in Eq. (3.11), this makes sense due to the fact that car curvature κ_v is dependent on the longitudinal velocity. As the velocity profile changes by changing μ, the same holds for car curvature κ_v. A value of μ less than 1 seemed sufficient to capture velocity change in curves without effecting the optimal trajectory.

Unfortunately, real-time feasibility by retrieving time information according to Eq. (3.14) has not been tested due to time constraints. However, since the optimization problem is formulated in a convex framework and vehicle dynamics are successively linearized and discretized, efficient run times can be expected in effort of real-time feasibility (Frasch et al., 2013). The trade-off between computation times (i.e. real-time feasibility) and complexity of the problem is common for MPC formulations. This thesis focused on efficient computation times rather than deploying a complex vehicle model.

Chapter 8

Conclusion

In this thesis, a Spatial-Based MPC algorithm is proposed for autonomous racing. A main method within the proposed algortihm is the reformulation of the time dynamics into spatial dynamics. The biggest advantage of the spatial dynamics is the allowance of formulating obstacles as constraints on the driving environment. The controller aims at maximizing the progress (i.e. minimizing travel time) along the center-line of a predefined track, while ensuring smoothness of the trajectory. The problem is formulated as a convex optimization problem by linearizing the vehicle dynamics successively around previous predicted optimal solutions of the MPC scheme.

Formulation of the problem in this fashion resulted in extremely efficient computation times, making real-time implementation more achievable. The proposed spatially vehicle dynamics allows for natural formulation of obstacles and track conditions. Consequently, speed dependence in the formulation of vehicle dynamics is eliminated and modeled as independent state. By introducing friction coefficient μ, the lateral vehicle dynamics seemed to be more affected by the longitudinal vehicle dynamics as μ adopted higher values. Friction coefficient μ was able to capture the vehicle dynamics more realistically, especially in the curvy track segments.

On the other hand, vehicle dynamics at the limit of handling are not incorporated resulting in solutions that might not be compatible with reality. Especially within the MPC method, the trade-off between computational complexity and real-time feasibility/implementation is prominent. The proposed algorithm in this thesis chose for efficient computation over complex vehicle dynamics.

The proposed MPC algorithm can be considered robust with respect to multiple variations. The Receding Horizon fashion of modeling, inherent in MPC, has a natural robustness against uncertainties and disturbances in the environment by optimizing the problem every sample step Δ_s over the predicted horizon N. The controller was tested with step- and horizon lengths leading to feasible results. Moreover, multiple tracks were employed and simulated under different initial conditions.

Lastly, overtaking manoeuvres are implemented by changing spatial boundary conditions in the driving environment. By doing so, both static and dynamic obstacles can be formulated depending on the length of spatial bounds. Depending on the beforehand determined obstacle paths, the controller calculates the optimal trajectory along the obstacles. It was noticed that the MPC controller was able to perform robust overtaking manoeuvres in the presence of multiple race opponents.

Chapter 9

Future Research

This thesis mainly focused on trajectory generation by considering a rather simple vehicle model. A interesting research direction would be to deploy a more complex dynamic model to simulate the behaviour. Tire dynamics (which plays a big role in racing) could be included by deploying such a complex vehicle model. In many researches, path planning and control is done in separately. The path planning part is often sufficiently executed by means of a rather simple vehicle model (such as the kinematic bicycle model) and the control part by considering a more complex model. This entails implementing a two-folded approach in stead of an integrated plan- and control algorithm. It would be interesting to see what the consequent advantages and disadvantages of such an hierarchical approach would be in comparison to the integrated approach.

A second future project in mind is to simulate the behaviour in software extensively used for vehicle control like CarSim, ARCADO, ForcesPro. Programs like these are extensively used within the field of vehicle control to more accurately model vehicle dynamics. As for the DTPA lab, an alternative could be to deploy a representative Ackermann Steering wheel robot into Gazebo/ROS. By doing so, the behaviour of the controller could be simulated more efficiently in compatible software such as the ones mentioned above.

Lastly, an interesting direction would be to formulate the problem as more complex (other than Convex), i.e. Quadratic programming in order to model in other vehicle dynamically as state. Modeling the problem as convex optimization problem is very time-efficient, but could be less accurate as well. Exploring real-time feasibility by changing dynamics back to time could be another option within this last future research direction.

Appendix A

Convexity Definitions (Princeton University, 2020)

Definition A.0.1. *A* set $\Omega \subseteq \mathbb{R}^n$ is convex, if for all $x, y \in \Omega$ and $\forall \lambda \in [0,1]$

$$\lambda x + (1-\lambda)y \in \Omega$$

A point of the form $\lambda x + (1-\lambda)y, \lambda \in [0,1]$ is called a convex combination of x and y. As λ varies between [0,1] , a "line segment" is being formed between x and y as shown in Figure A.1.

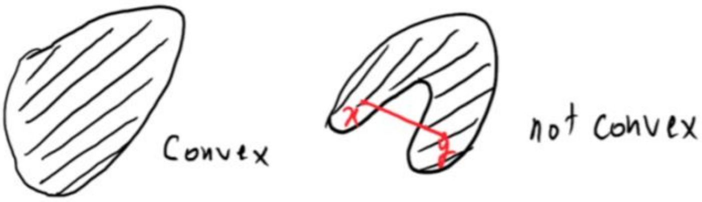


FIGURE A.1: Convexity of sets

Definition A.0.2. A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if its domain dom(f) is a convex set and for all $x, y \in \text{dom}(f)$ and $\forall \lambda \in [0,1]$, we have

$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y)$$

In words, this means that if we take any two points x, y, then f evaluated at any convex combination of these two points should be no larger than the same convex combination of $f(x)$ and $f(y)$. Geometrically, the line segment connecting $(x, f(x))$ to $(y, f(y))$ must sit above the graph of f as in Figure A.2.

Definition A.0.3. A function, $f : \mathbb{R}^n \to \mathbb{R}$, is linear if $f(x+y) = f(x) + f(y)$ and $f(\lambda x) = \lambda f(x)$ for all $x, y \in U$ such that either $x + y$ or $xy \in U$ respectively.

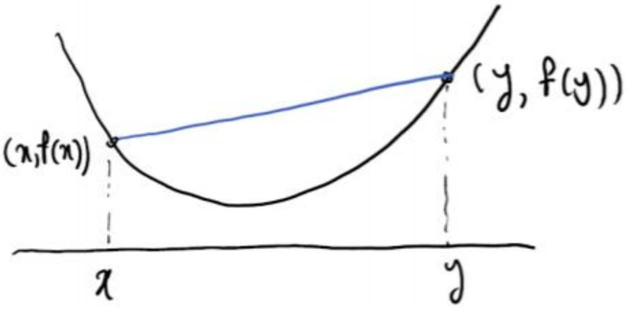


FIGURE A.2: Illustration of definition of convexity of function

Following Definitions A.0.3 and A.0.2, it can be concluded that linear functions are per definition convex. By setting $f(\lambda x + \lambda y) = \lambda f(x) + \lambda f(y)$ in Definition A.0.3, it follows Definition A.0.2 also $\forall \lambda \in [0, 1]$. These properties are fundamental for the considered control objective in Section 4.2.

Appendix B

MPC Matrix Derivation

B.1 Taylor Series

Since Taylor approximations are key in deriving the objective function for this thesis, a general formulation is given in this appendix. Consider a nonlinear function $f(\zeta)$ of one variable ζ, the Taylor series around linearization point $\zeta = a$ is given as

$$f(\zeta) = \sum_{k=0}^{\infty} \frac{f^{(k)}(a)}{k!}(\zeta - a)^k = \underbrace{\underbrace{f(a) + f'(a)(\zeta - a)}_{1^{st} order} + \frac{f''(a)}{2!}(\zeta - a)^2}_{2^{nd} order} + \mathcal{O} \tag{B.1}$$

Where linear and quadratic approximations are indicated as respectively first- and second order approximations of the Taylor series. Higher order terms are denoted as $\mathcal{O}$. When the function consists of more variables such as

$$\frac{dy}{dt} = f(y, u) \tag{B.2}$$

,

then the multi-variable Taylor series, considering two variables y and u, around $(\bar{y}, \bar{u})$ is

$$\frac{dy}{dt} = f(y, u) \approx f(\bar{y}, \bar{u}) + \left.\frac{\partial f}{\partial y}\right|_{\bar{y},\bar{u}} (y - \bar{y}) + \left.\frac{\partial f}{\partial u}\right|_{\bar{y},\bar{u}} (u - \bar{u}). \tag{B.3}$$

To simplify the final linearized expression, deviation variable are denoted as $y' = y - \bar{y}$ and $u' = u - \bar{u}$, resulting in the following approximation:

$$\frac{dy'}{dt} = \gamma_1 y' + \gamma_2 u'. \tag{B.4}$$

The values of constants γ_1 and γ_2 are the partial derivatives of $f(y, u)$ at steady state conditions:

$$\alpha = \left.\frac{\partial f}{\partial y}\right|_{\bar{y},\bar{u}} \qquad \beta = \left.\frac{\partial f}{\partial u}\right|_{\bar{y},\bar{u}} \tag{B.5}$$

B.2 Linearization

This appendix is devoted to a detailed description of the matrices used in the MPC formulation of chapter 4.

The linearization is done by Taylor approximating the nonlinear dynamics around the estimated optimal solutions (i.e. the solutions of one MPC iteration) given by $(\bar{\zeta}, \bar{u})$. In general notation, this yields the following:

$$\begin{aligned} f(\zeta,u) &\approx f\left(\bar{\zeta},\bar{u}\right) + \left.\frac{\partial f(\zeta,u)}{\partial z}\right|_{\bar{\zeta},\bar{u}} \left(\zeta-\bar{\zeta}\right) + \left.\frac{\partial f(\zeta,u)}{\partial u}\right|_{\bar{\zeta},\bar{u}} (u-\bar{u}) \Leftrightarrow \\ \Leftrightarrow f(\zeta,u) &\approx f\left(\bar{\zeta},\bar{u}\right) + A_c\left(\zeta-\bar{\zeta}\right) + B_c\left(u-\bar{u}\right) \end{aligned} \tag{B.6}$$

.

This allows to write (B.6) in the following compact linear form:

$$\dot{\zeta} = A_c\zeta + B_c u + H_c \tag{B.7}$$

Where continuous matrices A_c, B_c and constant matrix H_c are given for the problem at hand in (B.8) and (B.9)

$$\dot{\zeta} = \begin{bmatrix} \frac{\partial f_1}{\partial E_y} & \frac{\partial f_1}{\partial E_\psi} & \frac{\partial f_1}{\partial v_x} & \frac{\partial f_1}{\partial \kappa_v} \\ \frac{\partial f_2}{\partial E_y} & \frac{\partial f_2}{\partial E_\psi} & \frac{\partial f_2}{\partial v_x} & \frac{\partial f_2}{\partial \kappa_v} \\ \frac{\partial f_3}{\partial E_y} & \frac{\partial f_3}{\partial E_\psi} & \frac{\partial f_3}{\partial v_x} & \frac{\partial f_3}{\partial \kappa_v} \\ \frac{\partial f_4}{\partial E_y} & \frac{\partial f_4}{\partial E_\psi} & \frac{\partial f_4}{\partial v_x} & \frac{\partial f_4}{\partial \kappa_v} \end{bmatrix} \left(\zeta-\bar{\zeta}\right) + \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{a} & \frac{\partial f_4}{\partial C} \end{bmatrix} (u-\bar{u}) + f(\bar{\zeta},\bar{u}) \tag{B.8}$$

Which can be rewritten to make it discretizable as follows:

$$\begin{aligned} \dot{\zeta} = &\underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial E_y} & \frac{\partial f_1}{\partial E_\psi} & \frac{\partial f_1}{\partial v_x} & \frac{\partial f_1}{\partial \kappa_v} \\ \frac{\partial f_2}{\partial E_y} & \frac{\partial f_2}{\partial E_\psi} & \frac{\partial f_2}{\partial v_x} & \frac{\partial f_2}{\partial \kappa_v} \\ \frac{\partial f_3}{\partial E_y} & \frac{\partial f_3}{\partial E_\psi} & \frac{\partial f_3}{\partial v_x} & \frac{\partial f_3}{\partial \kappa_v} \\ \frac{\partial f_4}{\partial E_y} & \frac{\partial f_4}{\partial E_\psi} & \frac{\partial f_4}{\partial v_x} & \frac{\partial f_4}{\partial \kappa_v} \end{bmatrix}}_{A_c} \zeta + \underbrace{\begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix}}_{B_c} u \\ &\underbrace{+ f(\bar{\zeta},\bar{u}) - \begin{bmatrix} \frac{\partial f_1}{\partial E_y} & \frac{\partial f_1}{\partial E_\psi} & \frac{\partial f_1}{\partial v_x} & \frac{\partial f_1}{\partial \kappa_v} \\ \frac{\partial f_2}{\partial E_y} & \frac{\partial f_2}{\partial E_\psi} & \frac{\partial f_2}{\partial v_x} & \frac{\partial f_2}{\partial \kappa_v} \\ \frac{\partial f_3}{\partial E_y} & \frac{\partial f_3}{\partial E_\psi} & \frac{\partial f_3}{\partial v_x} & \frac{\partial f_3}{\partial \kappa_v} \\ \frac{\partial f_4}{\partial E_y} & \frac{\partial f_4}{\partial E_\psi} & \frac{\partial f_4}{\partial v_x} & \frac{\partial f_4}{\partial \kappa_v} \end{bmatrix} \bar{\zeta} - \begin{bmatrix} \frac{\partial f_1}{\partial a} & \frac{\partial f_1}{\partial C} \\ \frac{\partial f_2}{\partial a} & \frac{\partial f_2}{\partial C} \\ \frac{\partial f_3}{\partial a} & \frac{\partial f_3}{\partial C} \\ \frac{\partial f_4}{\partial a} & \frac{\partial f_4}{\partial C} \end{bmatrix} \bar{u}}_{H_c} \end{aligned} \tag{B.9}$$

Where H_c is constant and, therefore, not directly part of the discretization. The discretization process is further explained in the next section.

B.3 Discretization

The discretization of the continuous linearized system as described in appendix B.2, is done using the Zero-Order-Hold (ZOH) method. The ZOH method converts the continuous space equations to discrete space equations by holding each sample value for one sample interval Δ_s, where it is common to assume constant control signals between the samples (Åström and Wittenmark, 2013). By doing so, the linear model from (B.7) can be written as an LTV-model (Linear Space Varying in this case) as explained in chapter 4 in the following way:

$$\zeta(k+1) = A_d^{\text{ZOH}}(k)\zeta(k) + B_d^{\text{ZOH}}(k)u(k) + H_d^{\text{ZOH}}(k), k \geqslant 0. \tag{B.10}$$

These discretized matrices can be derived following

$$\begin{aligned} A_d^{\text{ZOH}}(k) &= \exp\left(A_c(k)\Delta_s\right) \\ B_d^{ZOH}(k) &= \left(\int_0^{\Delta_s} \exp\left(A_c(k)\tau\right) \mathrm{d}\tau\right) B_c(k) = A^{-1}\left(A_d - I\right) B_c \\ H_d^{\text{ZOH}}(k) &= H_c(k) \end{aligned} \tag{B.11}$$

,

Where the discretized matrices are obtained by successively applying B.6 each step, using the discretized version of the reference $(\bar{\zeta}(k), \bar{u}(k))$.

Exact discretization following (B.11) is possible if and only if A_c is nonsingular (i.e. invertible). In case of singularity of A_c, the following property (DeCarlo, Zak, and Matthews, 1988) can be used to get an auxiliary matrix M for every sampling instant k:

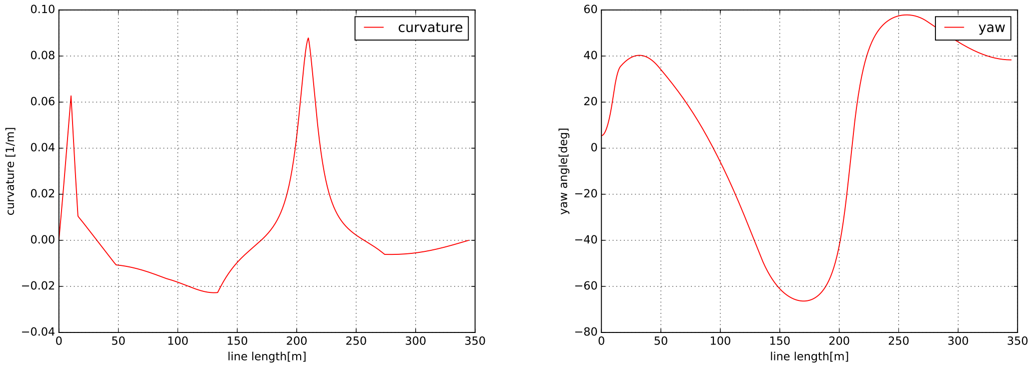
$$M(k) = \exp\left(\begin{bmatrix} A_c & B_c \\ 0 & 0 \end{bmatrix} \Delta_s\right) = \begin{bmatrix} A_d^{\text{ZOH}} & B_d^{\text{ZOH}} \\ 0 & I \end{bmatrix} \tag{B.12}$$

During the simulations, Eq. (B.12) seemed to give the more robust results in stability. Therefore, this approach of finding the discrete matrices is used. Recall that H_c in Eq. (B.9) is constant and, therefore, not taken directly into the discretization. $H_d^{\text{ZOH}}(k)$ is approximated as H_c, since H_c contains solutions generated from the discretized matrices A_d^{ZOH} and B_d^{ZOH} as in Eq. (B.12).

Appendix C

Additional track information

C.1 Track 1: S-curve

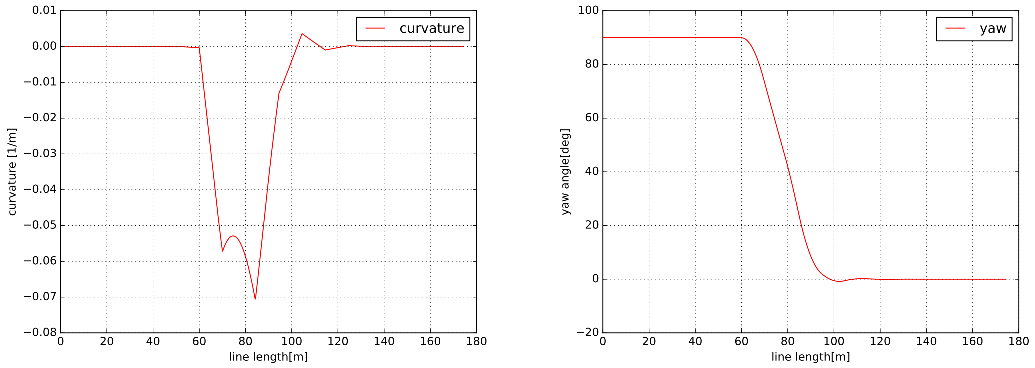


(A) Track curvature S-curve

(B) Yaw of track S-curve

FIGURE C.1: Track information S-curve

C.2 Track 2: 90-degrees curve

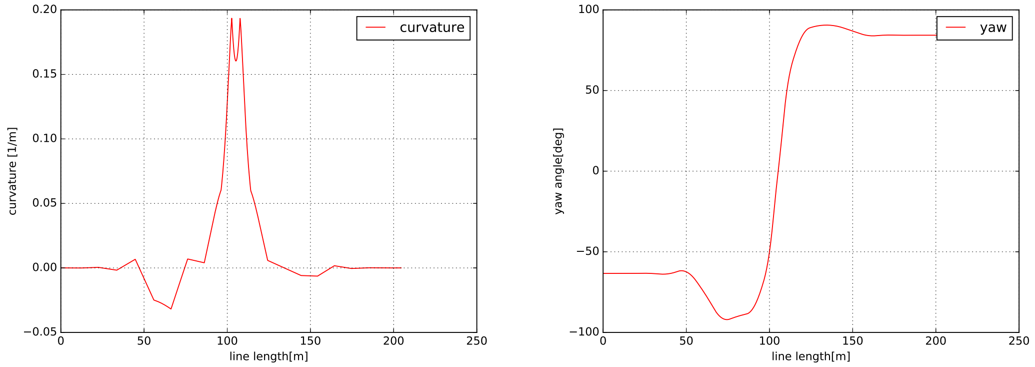


(A) Track curvature 90-degrees curve

(B) Yaw of track 90-degrees curve

FIGURE C.2: Track information 90-degrees curve

C.3 Track 3: Hairpin curve



(A) Track curvature hairpin curve

(B) Yaw of track hairpin curve

FIGURE C.3: Track information hairpin curve

Appendix D

Trajectories for changing friction coefficient μ

D.1 change in μ on S-curve

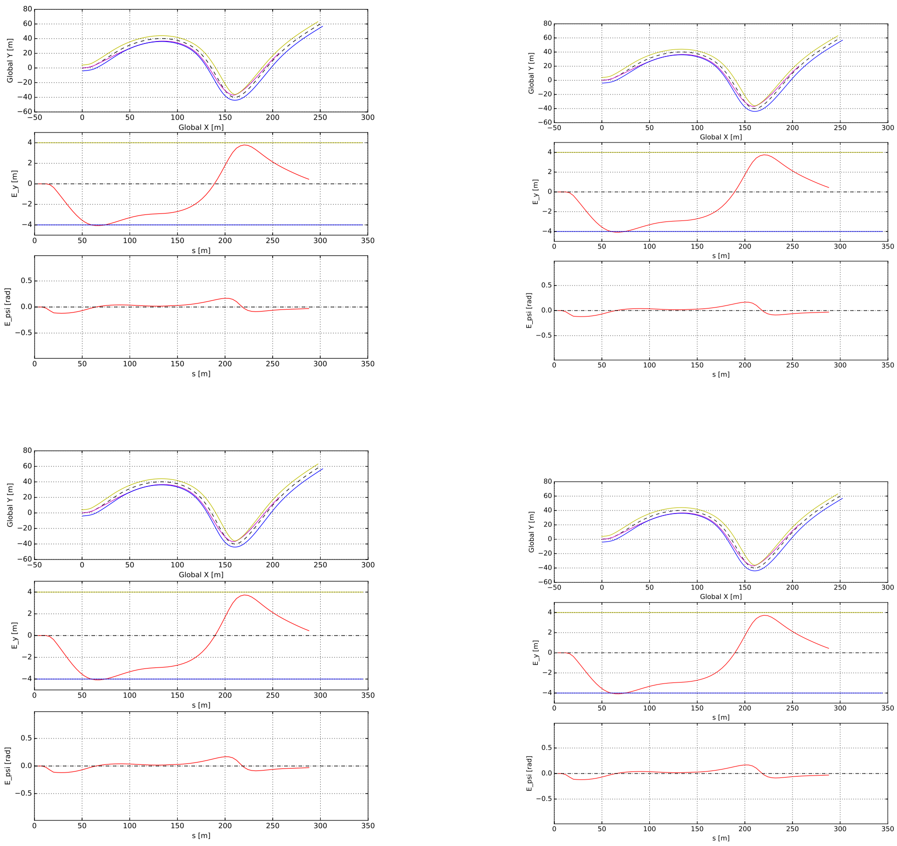


FIGURE D.1: From left to right: $\mu = 0, \mu = 0.5, \mu = 0.8, \mu = 1$

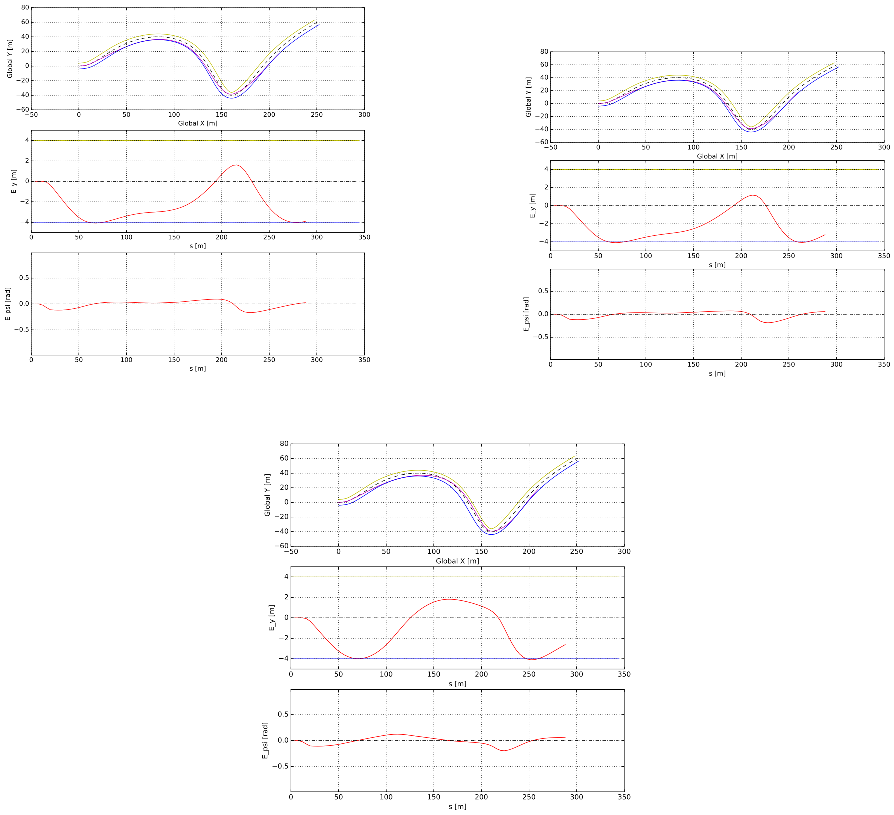


FIGURE D.2: From left to right: $\mu = 2, \mu = 3, \mu = 10$

D.2 Change in μ on hairpin curve

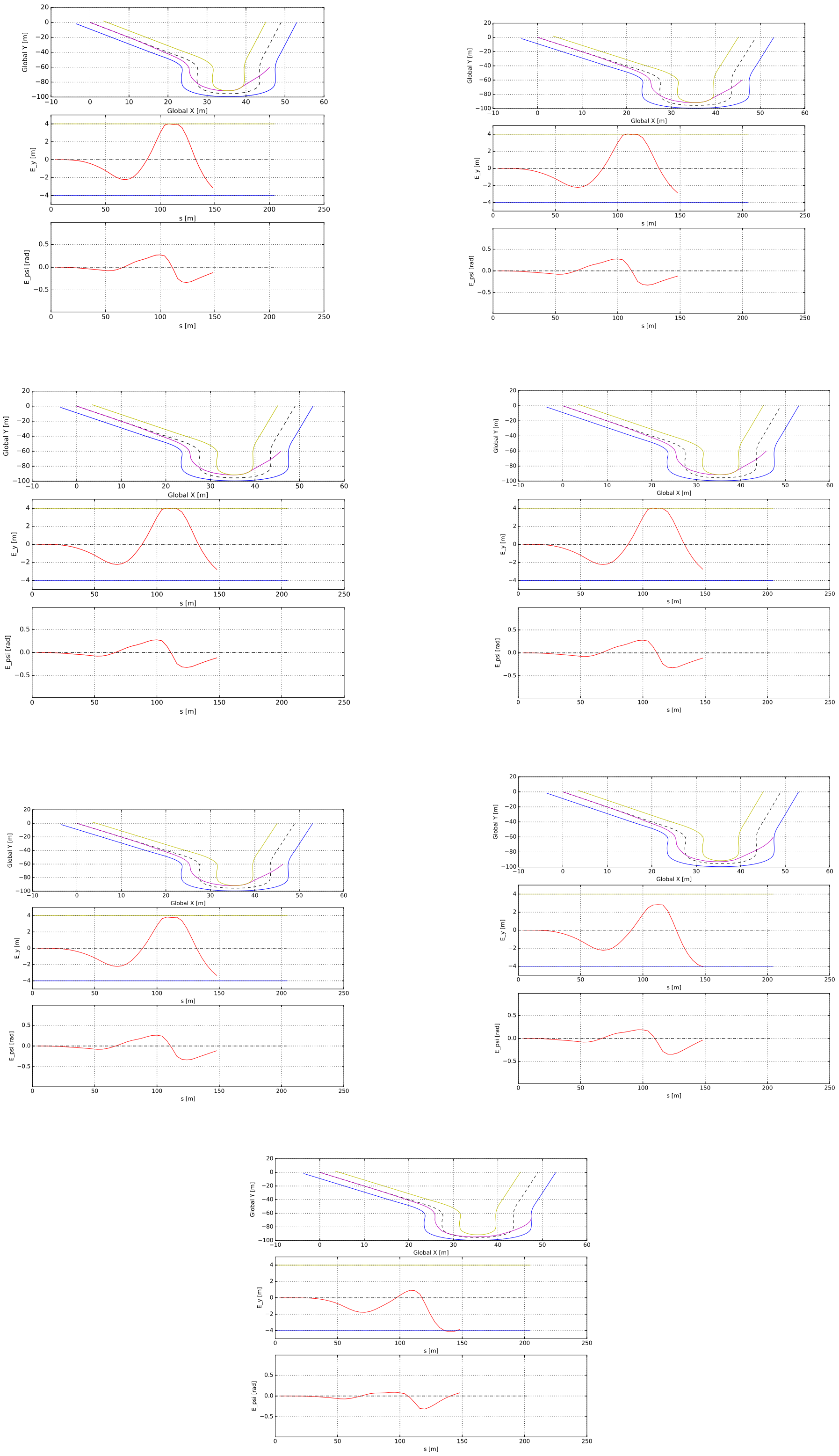
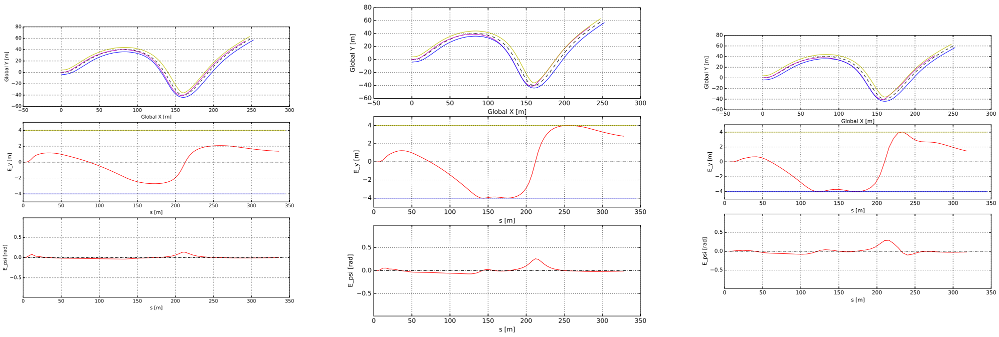


FIGURE D.3: From left to right: $\mu = 0, \mu = 0.5, \mu = 0.8, \mu = 1$, $\mu = 2, \mu = 3, \mu = 10$

Appendix E

Varying Horizon Lengths N and Step Lengths Δs

E.1 Constant $N = 5$ and varying Δ_s



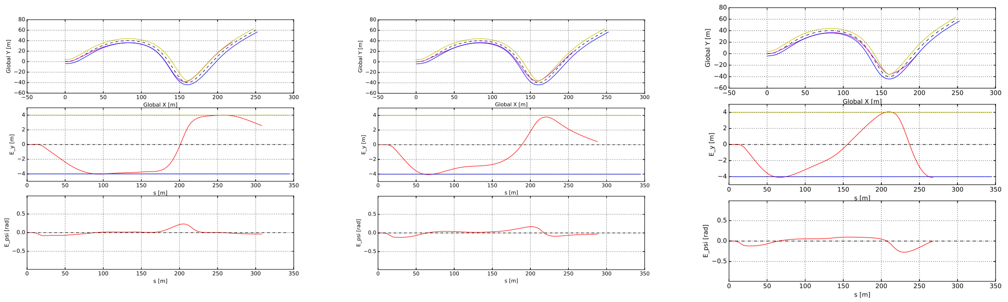
(A) $N = 5, \Delta_s = 2$

(B) $N = 5, \Delta_s = 4$

(C) $N = 5, \Delta_s = 6$

FIGURE E.1: S-curve trajectory with constant $N = 5$ and changing Δ_s

E.2 Constant $\Delta_s = 4$ and varying N



(A) $N = 10, \Delta_s = 4$

(B) $N = 15, \Delta_s = 4$

(C) $N = 20, \Delta_s = 4$

FIGURE E.2: S-curve trajectory with constant $N = 5$ and changing Δ_s

Bibliography

Allgöwer, Frank and Alex Zheng (2012). *Nonlinear model predictive control*. Vol. 26. Birkhäuser.

Amer, Noor Hafizah et al. (2017). "Modelling and Control Strategies in Path Tracking Control for Autonomous Ground Vehicles: A Review of State of the Art and Challenges". In: *Journal of Intelligent & Robotic Systems* 86.2, pp. 225–254. ISSN: 1573-0409. DOI: 10.1007/s10846-016-0442-0. URL: https://doi.org/10.1007/s10846-016-0442-0.

Amoozadeh, Mani et al. (2015). "Platoon management with cooperative adaptive cruise control enabled by VANET". In: *Vehicular Communications* 2.2, pp. 110 –123. ISSN: 2214-2096. DOI: https://doi.org/10.1016/j.vehcom.2015.03.004. URL: http://www.sciencedirect.com/science/article/pii/S2214209615000145.

Anavatti, S. G., S. L. Francis, and M. Garratt (2015). "Path-planning modules for Autonomous Vehicles: Current status and challenges". In: *2015 International Conference on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation (ICAMIMIA)*, pp. 205–214. DOI: 10.1109/ICAMIMIA.2015.7508033.

Angelov, P. (Apr. 2012). "Sense and Avoid in UAS: Research and Applications". In: *Sense and Avoid in UAS: Research and Applications*, pp. 265–294. DOI: 10.1002/9781119964049.ch10.

Åström, Karl J and Björn Wittenmark (2013). *Computer-controlled systems: theory and design*. Courier Corporation.

Bevan, Geraint P, H Gollee, and J O'reilly (2010). "Trajectory generation for road vehicle obstacle avoidance using convex optimization". In: *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* 224.4, pp. 455–473.

Biagiotti, Luigi and Claudio Melchiorri (2008). *Trajectory planning for automatic machines and robots*. Springer Science & Business Media.

Bonab, S. A. and A. Emadi (2019). "Optimization-based Path Planning for an Autonomous Vehicle in a Racing Track". In: *IECON 2019 - 45th Annual Conference of the IEEE Industrial Electronics Society*. Vol. 1, pp. 3823–3828. DOI: 10.1109/IECON.2019.8926856.

Brookhuis, Karel A, Dick De Waard, and Wiel H Janssen (2019). "Behavioural impacts of advanced driver assistance systems–an overview". In: *European Journal of Transport and Infrastructure Research* 1.3.

Buyval, A. et al. (2017). "Deriving overtaking strategy from nonlinear model predictive control for a race car". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2623–2628. DOI: 10.1109/IROS.2017.8206086.

Cafiso, Salvatore and Alessandro Di Graziano (May 2012). "Evaluation of the effectiveness of ADAS in reducing multi-vehicle collisions". In: *Int. J. of Heavy Vehicle Systems* 19, pp. 188 –206. DOI: 10.1504/IJHVS.2012.046834.

De Luca, Alessandro, Giuseppe Oriolo, and Claude Samson (1998). “Feedback control of a nonholonomic car-like robot”. In: *Robot motion planning and control*. Springer, pp. 171–253.

DeCarlo, Raymond A, Stanislaw H Zak, and Gregory P Matthews (1988). “Variable structure control of nonlinear multivariable systems: a tutorial”. In: *Proceedings of the IEEE* 76.3, pp. 212–232.

Diamond, Steven and Stephen Boyd (2016). “CVXPY: A Python-Embedded Modeling Language for Convex Optimization”. In: *Journal of Machine Learning Research* 17.83, pp. 1–5.

Dixit, Shilp et al. (Mar. 2018). “Trajectory planning and tracking for autonomous overtaking: State-of-the-art and future prospects”. In: *Annual Reviews in Control*. DOI: 10.1016/j.arcontrol.2018.02.001.

Falcone, Paolo et al. (2008). “Linear time-varying model predictive control and its application to active steering systems: Stability analysis and experimental validation”. In: *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal* 18.8, pp. 862–875.

Forbes (2019). *Millions Of Jobs Have Been Lost To Automation. Economists Weigh In On What To Do About It*. URL: https://www.forbes.com/sites/amysterling/2019/06/15/automated-future/#5b1e6a1d779d (visited on 10/10/2019).

Frasch, Janick et al. (July 2013). “An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles”. In: pp. 4136–4141. DOI: 10.23919/ECC.2013.6669836.

Gajbhiye, Sneha and Ravi N. Banavar (2016). “Geometric tracking control for a nonholonomic system: a spherical robot”. In: *IFAC-PapersOnLine* 49.18. 10th IFAC Symposium on Nonlinear Control Systems NOLCOS 2016, pp. 820 –825. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2016.10.267. URL: http://www.sciencedirect.com/science/article/pii/S240589631631847X.

Guo, H. et al. (2018). “Simultaneous Trajectory Planning and Tracking Using an MPC Method for Cyber-Physical Systems: A Case Study of Obstacle Avoidance for an Intelligent Vehicle”. In: *IEEE Transactions on Industrial Informatics* 14.9, pp. 4273–4283. DOI: 10.1109/TII.2018.2815531.

Hiraoka, Toshihiro, Osamu Nishihara, and Hiromitsu Kumamoto (2009). “Automatic path-tracking controller of a four-wheel steering vehicle”. In: *Vehicle System Dynamics* 47.10, pp. 1205–1227. DOI: 10.1080/00423110802545919. eprint: https://doi.org/10.1080/00423110802545919. URL: https://doi.org/10.1080/00423110802545919.

Hoffmann, G. M. et al. (2007). “Autonomous Automobile Trajectory Tracking for Off-Road Driving: Controller Design, Experimental Validation and Racing”. In: *2007 American Control Conference*, pp. 2296–2301. DOI: 10.1109/ACC.2007.4282788.

Junmin Wang, J. Steiber, and B. Surampudi (2008). “Autonomous ground vehicle control system for high-speed and safe operation”. In: *2008 American Control Conference*, pp. 218–223. DOI: 10.1109/ACC.2008.4586494.

Kala, Rahul and Kevin Warwick (2015). “Motion Planning of Autonomous Vehicles on a Dual Carriageway without Speed Lanes”. In: *Electronics* 4.1, pp. 59–81. ISSN: 2079-9292. DOI: 10.3390/electronics4010059. URL: https://www.mdpi.com/2079-9292/4/1/59.

Katrakazas, Christos et al. (2015). “Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions”. In: *Transportation Research Part C: Emerging Technologies* 60, pp. 416 –442. ISSN: 0968-090X. DOI: https://doi.org/10.1016/j.trc.2015.09.011. URL: http://www.sciencedirect.com/science/article/pii/S0968090X15003447.

Katriniok, A. and D. Abel (2011). “LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics”. In: *2011 50th IEEE Conference on Decision and Control and European Control Conference*, pp. 6828–6833. DOI: 10.1109/CDC.2011.6161257.

Kissai, Moad et al. (Apr. 2019). “Adaptive Robust Vehicle Motion Control for Future Over-Actuated Vehicles”. In: *Machines* 7, p. 26. DOI: 10.3390/machines7020026.

Kong, J. et al. (2015). “Kinematic and dynamic vehicle models for autonomous driving control design”. In: *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1094–1099. DOI: 10.1109/IVS.2015.7225830.

Kritayakirana, Krisada and J. Christian Gerdes (2012). “Using the centre of percussion to design a steering controller for an autonomous race car”. In: *Vehicle System Dynamics* 50.sup1, pp. 33–51. DOI: 10.1080/00423114.2012.672842. eprint: https://doi.org/10.1080/00423114.2012.672842. URL: https://doi.org/10.1080/00423114.2012.672842.

Lee, Taeyoung, Melvin Leok, and N Harris McClamroch (2010). “Geometric tracking control of a quadrotor UAV on SE (3)”. In: *49th IEEE conference on decision and control (CDC)*. IEEE, pp. 5420–5425.

Lima, P. F. et al. (2018). “Progress Maximization Model Predictive Controller”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1075–1082. DOI: 10.1109/ITSC.2018.8569647.

Liu, Jiechao et al. (Oct. 2014). “A Multi-Stage Optimization Formulation for MPC-Based Obstacle Avoidance in Autonomous Vehicles Using a LIDAR Sensor”. In: vol. 2. DOI: 10.1115/DSCC2014-6269.

Lucet, Eric, Roland Lenain, and Christophe Grand (2015). “Dynamic path tracking control of a vehicle on slippery terrain”. In: *Control Engineering Practice* 42, pp. 60 –73. ISSN: 0967-0661. DOI: https://doi.org/10.1016/j.conengprac.2015.05.008. URL: http://www.sciencedirect.com/science/article/pii/S0967066115000982.

Morin, Pascal and Claude Samson (2008). “Motion control of wheeled mobile robots.” In: *Springer handbook of robotics* 1, pp. 799–826.

NHTSA (2018). *MAutomated Driving Systems AV 3.0*. URL: https://www.nhtsa.gov/vehicle-manufacturers/automated-driving-systems#automated-driving-systems-av-20 (visited on 10/15/2019).

Ozguner, Umit, Tankut Acarman, and Keith Redmill (2011). *Autonomous ground vehicles*. Artech House.

Pascual, Carrasco and Juan José Marcuello Pablo (2009). “Advanced driver assistance system based on computer vision using detection, recognition and tracking of road signs”. In:

Princeton University (2020). *Convex and Conic Optimization*. URL: http://aaa.princeton.edu/orf523.

RDW (2018). *Safe and reliable on the road - The mission, vision and strategy of the RDW*. URL: https://www.rdw.nl/-/media/rdw/rdw/pdf/sitecollectiondocuments/over-rdw/brochures-en-folders/rdw_strategy_en_def.pdf (visited on 10/17/2019).

SAE (2018). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. URL: hhttps://www.sae.org/standards/content/j3016_201806/ (visited on 10/10/2019).

Schwarting, Wilko, Javier Alonso-Mora, and Daniela Rus (2018). “Planning and Decision-Making for Autonomous Vehicles”. In: *Annual Review of Control, Robotics, and Autonomous Systems* 1.1, pp. 187–210. DOI: 10.1146/annurev-control-060117-105157. URL: https://doi.org/10.1146/annurev-control-060117-105157.

Schwarting, Wilko et al. (2017). “Safe nonlinear trajectory generation for parallel autonomy with a dynamic vehicle model”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.9, pp. 2994–3008.

Shahzad, Amir, Eric C Kerrigan, and George A Constantinides (2010). “A warm-start interior-point method for predictive control”. In:

Song, S. et al. (2015). “Autonomous Vehicle Control Near the Limit of Friction”. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 1008–1013. DOI: 10.1109/ITSC.2015.168.

Zhang, S. et al. (2013). “Dynamic Trajectory Planning for Vehicle Autonomous Driving”. In: *2013 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4161–4166. DOI: 10.1109/SMC.2013.709.