Master Thesis

Czech Technical University in Prague

F3 Faculty of Electrical Engineering
Department of Control Engineering

Driving Envelope Protection Using Nonlinear Model Predictive Control

Bc. Tomáš Velecký

Supervisor: Ing. Denis Efremov
Study Programme: Cybernetics and Robotics
Field of Study: Cybernetics and Robotics
January 2021

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Velecký Tomáš**   Personal ID number: **457241**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Control Engineering**

Study program: **Cybernetics and Robotics**

Branch of study: **Cybernetics and Robotics**

II. Master's thesis details

Master's thesis title in English:

Driving envelope protection using nonlinear model predictive control

Master's thesis title in Czech:

Vývoj řídicích systémů pro ochranu jízdní obálky pomocí nelineárního prediktivního řízení

Guidelines:

The goal of the thesis is to investigate driving envelope definition and suggest nonlinear model predictive control formulation
for driving envelope protection. Provide validation and verification on a car simulator.
1. Study driving envelope definition.
2. Study documentation for CasADi (tool for nonlinear optimization).
3. Adopt single-track model for design purposes.
4. Formulate and implement nonlinear model predictive control to protect boundaries defined by the driving envelope.
5. Provide validation and verification ride test using racing simulator "Life for Speed".

Bibliography / sources:

[1] Schramm, D., Hiller, M., & Bardini, R. (2014). Vehicle dynamics. Modeling and Simulation. Berlin, Heidelberg,151.
[2] Efremov, D., Klaučo, M., Haniš, T., & Hromčík, M. (2019) "Driving envelope definition and envelope protection using
model predictive control". Accepted on American Control Conference (ACC2020).
[3] Andersson, J. A., Gillis, J., Horn, G., Rawlings, J. B., & Diehl, M. (2019). CasADi: a software framework for nonlinear
optimization and optimal control. Mathematical Programming Computation, 11(1), 1-36.

Name and workplace of master's thesis supervisor:

Ing. Denis Efremov,   Department of Control Engineering,   FEE

Name and workplace of second master's thesis supervisor or consultant:

Ing. Tomáš Haniš, Ph.D.,   Department of Control Engineering,   FEE

Date of master's thesis assignment: **27.07.2020**   Deadline for master's thesis submission: _____

Assignment valid until:
by the end of winter semester 2021/2022

_____   _____   _____
Ing. Denis Efremov
Supervisor's signature

prof. Ing. Michael Šebek, DrSc.
Head of department's signature

prof. Mgr. Petr Páta, Ph.D.
Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others,
with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

_____   _____
Date of assignment receipt   Student's signature

Acknowledgements

I thank my supervisor Ing. Denis Efremov for the topic of this thesis and his almost surprising enthusiasm which was very helpful in the past year.

Further thanks go to Bc. Tomáš Twardzik and Bc. Adam Konopiský for their help with the software they created to connect and test with the Live for Speed simulator; to Ing. Martin Klaučo, PhD., for his advice about optimization; and to the guys from Embotech for their solver and free support.

Essential was the study environment I got from my family. Furthermore, I am grateful for the proofreading, which was done too far away.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Frýdek-Místek, 15th January 2021,

. . . .

iii

Abstract

This thesis presents an implementation of a nonlinear model predictive control (NMPC) strategy for a racing car. The strategy takes into account the reference from a human driver (i.e. from the steering wheel, the accelerator and brake pedals) and the current state of the vehicle, and outputs an optimized version of the driver's actions on the vehicle state. In this way, possible mistakes of the driver, who has only a limited and approximate knowledge of the vehicle state, are mitigated. Common mistakes which might be corrected by this strategy include driving commands leading to locking the wheels, traction loss, and loss of directional control. Although similar safety systems, such as the anti-lock braking system (ABS) already exist, the presented strategy attempts to replace multiple safety systems by using a single *full-time-full-authority control system.* The idea of using a full-time-full-authority control comes from the flight control industry; it appears that the automotive industry could advance in this area. For the purposes of NMPC design, a single-track model of a car was chosen. The optimization problem constraints stem from a previously defined vehicle *driving envelope.* The resulting computer program was generated by the FORCESPRO optimization software; it runs in real-time on a laptop and was tested with the Live for Speed racing simulator.

Keywords: nonlinear model predictive control, MPC, drive-by-wire, driving envelope, FORCESPRO, single-track vehicle model

Abstrakt

Tato diplomová práce prezentuje implementaci nelineárního prediktivního řízení (NMPC) pro závodní auto. Vytvořený kontrolér přijímá referenci od lidského řidiče (stav volantu, brzdového a plynového pedálu) a informaci o stavu vozidla; na výstupu dává vylepšenou verze reference od řidiče. Takto vylepšený řídicí signál má minimalizovat případné chyby v řízení lidského řidiče, který má pouze omezenou a přibližnou znalost o stavu vozidla a jeho dynamice. Obvyklou chybou v řízení auta, které lze takto předejít, je například uzamčení kol, ztráta trakce a ztráta kontroly nad směrem vozidla. Ačkoli bezpečnostní sytémy mající tuto funkci, tedy protiblokovací systém (ABS), a další již existují, představený kontrolér má za cíl nahradit více takovýchto bezpečnostních systémů jedním tzv. *full-time-full-authority* systémem. Myšlenka full-time-full-authority systému pochází z leteckého průmyslu a zdá se, že automobilový průmysl by z ní také mohl těžit. Pro účely návrhu NMPC byl zvolen jednostopý model auta. Omezení optimalizační úlohy pak plynou z nové definice *řídicí obálky.* Výsledný počítačový program byl vygenerován s použitím softwaru FORCESPRO; je schopen běhu na notebooku v reálném čase, a byl testován na závodním simulátoru Live for Speed.

Klíčová slova: nelineární prediktivní řízení, MPC, drive-by-wire, jízdní obálka, FORCESPRO, jednostopý model auta

Překlad názvu: Vývoj řídicích systémů pro ochranu jízdní obálky pomocí nelineárního prediktivního řízení

Contents

Figures

Tables

Chapter 1

Introduction

In past years, requirements for vehicle passenger safety have significantly increased. Various sensors are being developed with the goal of safe, economic, and ultimately, autonomous cars. Those conditions have led to the implementation of new and better safety systems. However, these systems still mostly work separately from each other.

1.1 Driving Envelope

Inspired by a common approach in airplane engineering, a *driving envelope* definition was recently stated. The driving envelope of a vehicle, simply said, is a part of the space of specific vehicle state variables in which the vehicle is steerable. By steerable, we mean the vehicle's ability to respond in the way a driver wants or expects. Opposed to moments of steerability are critical, possibly dangerous, situations due to the loss of the wheel traction such as spinning of the vehicle, locking the wheels, or fishtailing. The driving envelope definition and its protection using model predictive control was suggested in [1].

The referenced driving envelope definition employs the tire forces and the variables commonly used to model them. It sets linear bounds on the sideslip angles and slip ratios and enforces nonlinear inequalities on the tire forces. By the semantics of the restricted variable or variables, we group these restrictions in the *lateral envelope*, *logitudinal envelope*, and the *combined slip envelope.*

1.2 Control Strategy

In [1], driving envelope protection has been implemented using a linear model predictive control strategy which demands less computational power, is easier to formulate, and always converges. Unfortunaly, the linear strategy has some disadvantages, namely that it forces simplification of both the vehicle

model and the constraints, as the model and some of the driving envelope constraints are nonlinear. Although it is sometimes difficult to formulate nonlinear model predictive controllers, current computing power makes it generally possible to do so.

The goal of this thesis is therefore to suggest a nonlinear model predictive control strategy formulation for the driving envelope protection.
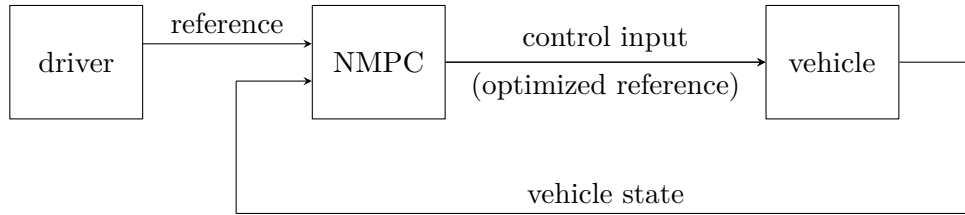
Figure 1.1: Diagram of the control loop

Diagram 1.1 shows a simplified top-level view of the proposed control loop. A human driver is driving the vehicle in a common way, i.e. using a steering wheel and the accelerator and brake pedals. (A vehicle with automatic gearbox is assumed.) The information from the mechanical controllers is taken as a reference for the NMPC. The state of the vehicle, namely its velocity, sideslip angle and yaw rate, is also an input of the NMPC block. The state is assumed to be measured on the vehicle; for the purposes of this thesis, it is taken from a vehicle dynamics simulator. The NMPC block estimates the wheel angular velocities the vehicle would gain if there was no MPC strategy. Taking this information, the proposed controller computes an optimized version of the reference and outputs it to the corresponding vehicle actuators.

▮ **1.3 Text Structure**

Chapter 2 recaps the thesis assignment and explains the workflow. Chapter 3 describes the adopted car model and Chapter 4 provides the driving envelope definition. As the model inputs do not correspond to the inputs of the racing simulator, an additional model dealing with wheel dynamics had to be created (Chapter 5). Further chapters treat the nonlinear model predictive control (NMPC) itself. First, the available software and existing approaches are summarized in Chapter 6. Second, the proposed NMPC formulation is explained in Chapter 7. The closing chapters introduce the simulation environment (Chapter 8) and demonstrate experimental results (Chapter 9).

Chapter 2

Objectives

The initial objectives of this thesis were:

1. To study the driving envelope definition.

2. To make use of a tool for nonlinear optimization, namely CasADi.

3. To adopt the single-track model of a vehicle.

4. To develop a nonlinear model predictive control (NMPC, nonlinear MPC) strategy which protects the boundaries given by the driving envelope.

5. Using the vehicle dynamics simulator Live for Speed, to compare the developed NMPC:

   - with existing linear MPC strategy,
   - with no additional control strategy.

Chapter 3

Single-Track Nonlinear Model of a Vehicle

The core of every *model predictive control* (MPC) is a model of the dynamics of the controlled system. For this thesis, the single-track model adopted from [2, 3] and used in [4] was chosen.

Both MPC strategies, the one developed in this thesis, and the one formulated in [1], are originally based on the same model. The difference lies in discretization — the linear MPC uses a linearized (differential) model in multiple linearization points, while the nonlinear MPC predicts the states in discrete time by integrating the continous model given the current states (which also change between the iterations of the solving process).

The single-track model presented in this chapter is identical to the model presented in [4] except for the number of steering angles; the simulator used here only offers cars with steerable front wheels. (Therefore δ_f is here δ, and δ_r is null.) Additional modifications are listed together with their reasoning in the following section (7.1).

3.1 Overview

The single-track model considers only the translation and rotation in a plane, using three states and three inputs. The model is depicted in Figure 3.1. On the sides, one can see the rear and front wheels (here on the left and right side, respectively). Model variables express motion in three points: the rear wheel, the front wheel, and the car's center of gravity (mass).

In the figure, angles are green, translational velocities are blue, forces are red, angular velocities are purple, and the dimensions are drawn in orange.
For tables summarizing the variables and parameters used in this chapter, please refer to Section 3.10. All single-track model variables are listed in Table 3.1. General single-track model parameters are listed in Table 3.2. (In Table 3.2, CG means the center of gravity.) Pacejka coefficients are in Table 3.3 and the coefficients used in the force scaling algorithm are in Table 3.4.
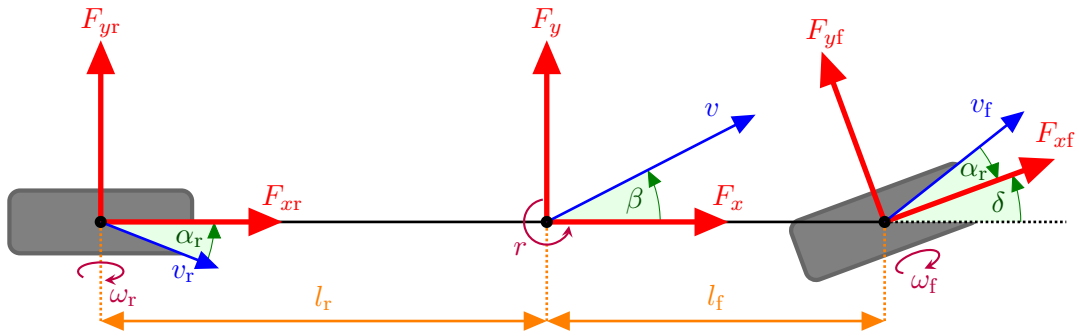
Figure 3.1: Geometric scheme of the single-track model

Model states are the car's sideslip angle β, velocity v, and yaw rate r. Model inputs are the steering angle δ, angular velocity of the front wheel ω_f and of the rear wheel ω_r.

3.2 Characteristics

The single-track model can be analyzed on the frequency spectrum from 0 to 2 Hz, according to [2].

The model was derived considering the following assumptions:

- Lift, pitch, and roll motions are neglected.

- All mass is located at a single point.

- Mass distribution on the axles is constant.

- Front and rear axles are represented as a single tire, resembling a bicycle. The contact points of tires to the surface lie on the same axis as the tires.

- Pneumatic trail and aligning torque resulting from the tire sideslip angle are neglected.

3.3 Model Parts

A block scheme of the model is shown in Figure 3.2. Following the block diagram from right to left, the vehicle is assumed to be a rigid body, the dynamics of which are determined by two force components and a rotational moment (**rigid body dynamics**). The forces, however, come from each wheel, and the direction is further influenced by the steering wheel (**steering angle projection**).

The source of forces acting upon the wheels is the contact of tires with the road surface. The tires are commonly modeled using the Pacejka Magic

Formula (**tire models**) and its help variables (**slip ratios**, **sideslip angles**). All help variables are computed using the velocity components of the wheels (**wheel kinematics**).

The wheels themselves are considered dynamic systems. This aspect will be discussed in Chapter 5 which describes a model of the wheels which provides the wheel angular velocities for the vehicle model described in this chapter.
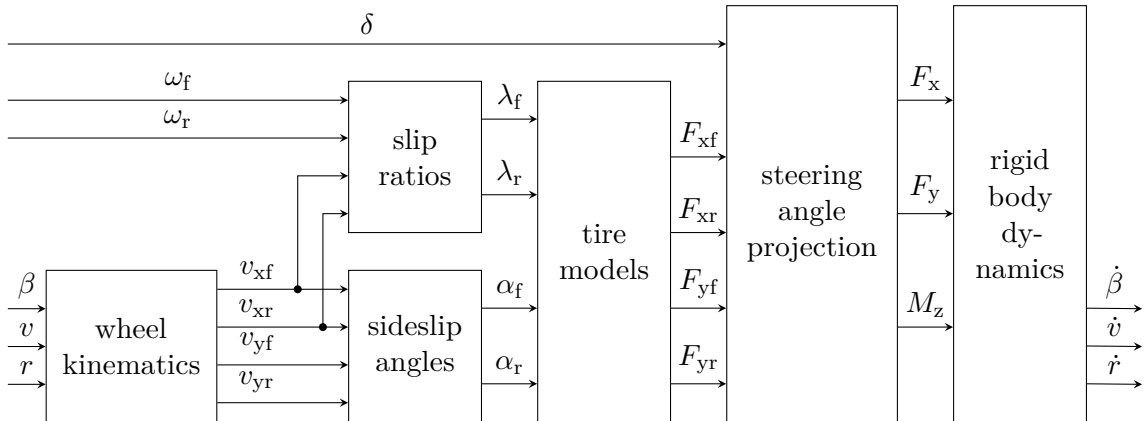
Figure 3.2: Block diagram of the single-track model

■ **3.4 Wheel Kinematics**

The wheel (translational) velocities depend both on the translational and the rotational movement of the vehicle body.

Let us assume we know both components of the translational velocity of the center of mass of the vehicle, v_x and v_y. Then, only the rotational movement is left to be expressed.

For the front wheel a circular rotation with radius l_f centered in the center of mass is considered. The vehicle body rotates with yaw rate (angular velocity) r. The tangential velocity is then $l_f r$. Its direction is in the y-axis of the vehicle coordinate system, so we can add it to v_y. We get the velocity vector of a point in the location of the front wheel,

$$\begin{bmatrix} v_x \\ v_y + l_f r \end{bmatrix}.$$

As the front wheel can be steered, we use a rotational matrix employing the angle δ. This gives the first row of equation 3.1.

Expressing the vehicle velocity components v_x and v_y in the terms of the single-track model states, $v_x = v \cdot \cos\beta$ and $v_y = v \cdot \sin\beta$, we have final expression for the front wheel velocity components.

$$
\begin{bmatrix} v_{xf} \\ v_{yf} \end{bmatrix} = \begin{bmatrix} \cos\delta & \sin\delta \\ -\sin\delta & \cos\delta \end{bmatrix} \cdot \begin{bmatrix} v_x \\ v_y + l_f r \end{bmatrix} =
$$
$$
= \begin{bmatrix} \cos\delta & \sin\delta \\ -\sin\delta & \cos\delta \end{bmatrix} \cdot \begin{bmatrix} v \cdot \cos\beta \\ v \cdot \sin\beta + l_f r \end{bmatrix}. \tag{3.1}
$$

The rear wheel is not steerable, therefore there is no rotational matrix. The reasoning for the rotational component is the same, except for a different distance of the wheel center from the center of mass: $l_f \rightarrow l_r$. The same rotation r acts here in the opposite direction:

$$
\begin{bmatrix} v_{xr} \\ v_{yr} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y - l_r r \end{bmatrix} = = \begin{bmatrix} v \cdot \cos\beta \\ v \cdot \sin\beta - l_r r \end{bmatrix}. \tag{3.2}
$$

▌ 3.5 Slip Ratios

Slip ratios are defined here as

$$
\lambda_f = \frac{\omega_f p - v_{xf}}{\max\left(|\omega_f p|, |v_{xf}|\right)}, \quad \lambda_r = \frac{\omega_r p - v_{xr}}{\max\left(|\omega_r p|, |v_{xr}|\right)}, \tag{3.3}
$$

where p is the rolling radius of a wheel.

They represent the relative difference of the actual angular velocity of the wheel (its rigid part) from the angular velocity it would have in a steady state, given by its translational velocity and wheel radius.

A slip ratio of a wheel is therefore usually *positive when accelerating*, in an extreme case, slipping; and it is usually *negative when braking*, having locking as an extreme case.

By used definition,
$$
\lambda_f \in [-1, 1], \quad \lambda_r \in [-1, 1]. \tag{3.4}
$$

▌ 3.6 Sideslip Angles

A sideslip angle is the angle between the x-axis of a wheel coordinate system (front direction) and the wheel's velocity vector (precisely, its projection to the horizontal plane). In the model, they are expressed as

$$
\alpha_f = -\arctan\frac{v_{yf}}{|v_{xf}|}, \quad \alpha_r = -\arctan\frac{v_{yr}}{|v_{xr}|}. \tag{3.5}
$$

The sideslip angle is positive when the wheel slips to the right and negative otherwise. This holds for moving forward and backward.

By definition: $\alpha_\mathrm{f} \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, $\alpha_\mathrm{r} \in [-\frac{\pi}{2}, \frac{\pi}{2}]$.

■ 3.7 Tire Models

This part computes the forces coming from the tires using twelve Pacejka coefficients and load forces as parameters.

Load forces are considered constant in this model:

$$F_{z,\mathrm{f}} = mg\frac{l_\mathrm{r}}{l_\mathrm{f} + l_\mathrm{r}}, \tag{3.6}$$

$$F_{z,\mathrm{r}} = mg\frac{l_\mathrm{f}}{l_\mathrm{f} + l_\mathrm{r}}, \tag{3.7}$$

where m is the mass of the vehicle and g is the gravitational acceleration.

The lateral Simplified Pacejka Magic Formula determines the lateral forces as a function of the sideslip angles:

$$F_{y\mathrm{f,raw}} = c_{\mathrm{D},y}F_{z\mathrm{f}} \sin\left(c_{\mathrm{C},y}\arctan\left(c_{\mathrm{B},y}\alpha_\mathrm{f} - c_{\mathrm{E},y}\left(c_{\mathrm{B},y}\alpha_\mathrm{f} - \arctan\left(c_{\mathrm{B},y}\alpha_\mathrm{f}\right)\right)\right)\right), \tag{3.8}$$

$$F_{y\mathrm{r,raw}} = c_{\mathrm{D},y}F_{z\mathrm{r}} \sin\left(c_{\mathrm{C},y}\arctan\left(c_{\mathrm{B},y}\alpha_\mathrm{r} - c_{\mathrm{E},y}\left(c_{\mathrm{B},y}\alpha_\mathrm{r} - \arctan\left(c_{\mathrm{B},y}\alpha_\mathrm{r}\right)\right)\right)\right), \tag{3.9}$$

where $c_{\mathrm{B},y},c_{\mathrm{C},y},c_{\mathrm{D},y},c_{\mathrm{E},y}$ are constants.

The logitudinal Simplified Pacejka Magic Formula is a function of the slip ratios determining the longitudinal forces:

$$F_{x\mathrm{f,raw}} = c_{\mathrm{D},x\mathrm{f}}F_{z\mathrm{f}} \sin\left(c_{\mathrm{C},x\mathrm{f}}\arctan\left(c_{\mathrm{B},x\mathrm{f}}\lambda_\mathrm{f} - c_{\mathrm{E},x\mathrm{f}}\left(c_{\mathrm{B},x\mathrm{f}}\lambda_\mathrm{f} - \arctan\left(c_{\mathrm{B},x\mathrm{f}}\lambda_\mathrm{f}\right)\right)\right)\right), \tag{3.10}$$

$$F_{x\mathrm{r,raw}} = c_{\mathrm{D},x\mathrm{r}}F_{z\mathrm{r}} \sin\left(c_{\mathrm{C},x\mathrm{r}}\arctan\left(c_{\mathrm{B},x\mathrm{r}}\lambda_\mathrm{r} - c_{\mathrm{E},x\mathrm{r}}\left(c_{\mathrm{B},x\mathrm{r}}\lambda_\mathrm{r} - \arctan\left(c_{\mathrm{B},x\mathrm{r}}\lambda_\mathrm{r}\right)\right)\right)\right), \tag{3.11}$$

where $c_{\mathrm{B},x\mathrm{f}}$, $c_{\mathrm{B},x\mathrm{r}}$, $c_{\mathrm{C},x\mathrm{f}}$, $c_{\mathrm{C},x\mathrm{r}}$, $c_{\mathrm{D},x\mathrm{f}}$, $c_{\mathrm{D},x\mathrm{r}}$, $c_{\mathrm{E},x\mathrm{f}}$, $c_{\mathrm{E},x\mathrm{r}}$ are constants.

Plots employing specific values of the parameters used in this thesis are shown in Figure 3.3.
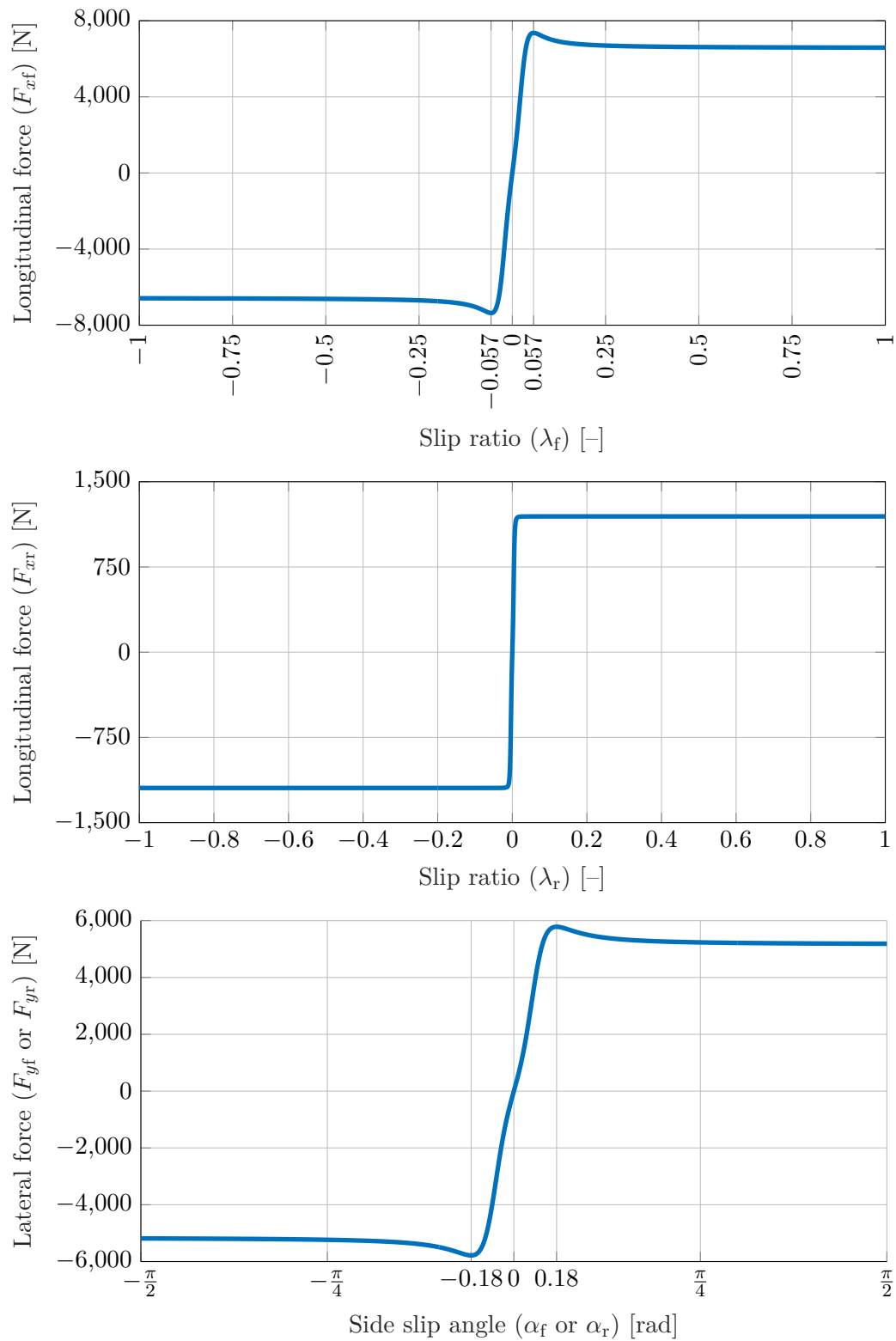
Figure 3.3: Force characteristics given by the Simplified Pacejka Magic Formula

9

The force components computed using Pacejka Magic Formula are only approximate. There is a limit on the combined force produced by a wheel, given by the normal force acting on the wheel, called the *traction ellipse*:

$$F_\mathrm{f} = \sqrt{\frac{F_{x\mathrm{f}}^2}{c_{\mathrm{D},x\mathrm{f}}^2} + \frac{F_{y\mathrm{f}}^2}{c_{\mathrm{D},y}^2}} \leq \mu F_{z\mathrm{f}}, \tag{3.12}$$

$$F_\mathrm{r} = \sqrt{\frac{F_{x\mathrm{r}}^2}{c_{\mathrm{D},x\mathrm{r}}^2} + \frac{F_{y\mathrm{r}}^2}{c_{\mathrm{D},y}^2}} \leq \mu F_{z\mathrm{r}}, \tag{3.13}$$

where μ is the friction coefficient of a road.

If necessary, a scaling algorithm is applied on the "raw" forces, originally described in [5]. First, the raw forces are "descaled":

$$\mu_{x\mathrm{f,act}} = \frac{F_{x\mathrm{f,raw}}}{F_{z\mathrm{f}}}, \quad \mu_{y\mathrm{f,act}} = \frac{F_{y\mathrm{f,raw}}}{F_{z\mathrm{f}}}, \tag{3.14}$$

$$\mu_{x\mathrm{r,act}} = \frac{F_{x\mathrm{r,raw}}}{F_{z\mathrm{r}}}, \quad \mu_{y\mathrm{r,act}} = \frac{F_{y\mathrm{r,raw}}}{F_{z\mathrm{r}}}, \tag{3.15}$$

and help variables are computed:

$$\beta_\mathrm{f}^* = \arccos \frac{|\lambda_\mathrm{f}|}{\sqrt{\lambda_\mathrm{f}^2 + \sin^2 \alpha_\mathrm{f}}}, \quad \beta_\mathrm{r}^* = \arccos \frac{|\lambda_\mathrm{r}|}{\sqrt{\lambda_\mathrm{r}^2 + \sin^2 \alpha_\mathrm{r}}}. \tag{3.16}$$

Taking more precise friction coefficients for each wheel:

$$\mu_{x\mathrm{f}} = \frac{1}{\sqrt{\left(\frac{1}{\mu_{x\mathrm{f,act}}}\right)^2 + \left(\frac{\tan \beta_\mathrm{f}^*}{c_{\mathrm{D},y}}\right)^2}}, \quad \mu_{y\mathrm{f}} = \frac{\tan \beta_\mathrm{f}^*}{\sqrt{\left(\frac{1}{c_{\mathrm{D},x\mathrm{f}}}\right)^2 + \left(\frac{\tan \beta_\mathrm{f}^*}{\mu_{y\mathrm{f,act}}}\right)^2}}, \tag{3.17}$$

$$\mu_{x\mathrm{r}} = \frac{1}{\sqrt{\left(\frac{1}{\mu_{x\mathrm{r,act}}}\right)^2 + \left(\frac{\tan \beta_\mathrm{r}^*}{c_{\mathrm{D},y}}\right)^2}}, \quad \mu_{y\mathrm{r}} = \frac{\tan \beta_\mathrm{r}^*}{\sqrt{\left(\frac{1}{c_{\mathrm{D},x\mathrm{r}}}\right)^2 + \left(\frac{\tan \beta_\mathrm{r}^*}{\mu_{y\mathrm{r,act}}}\right)^2}}, \tag{3.18}$$

the final forces yield

$$F_{x\mathrm{f}} = \left|\frac{\mu_{x\mathrm{f}}}{\mu_{x\mathrm{f,act}}}\right| \cdot F_{x\mathrm{f,raw}}, \quad F_{y\mathrm{f}} = \left|\frac{\mu_{y\mathrm{f}}}{\mu_{y\mathrm{f,act}}}\right| \cdot F_{y\mathrm{f,raw}}, \tag{3.19}$$

$$F_{x\mathrm{r}} = \left|\frac{\mu_{x\mathrm{r}}}{\mu_{x\mathrm{r,act}}}\right| \cdot F_{x\mathrm{r,raw}}, \quad F_{y\mathrm{r}} = \left|\frac{\mu_{y\mathrm{r}}}{\mu_{y\mathrm{r,act}}}\right| \cdot F_{y\mathrm{r,raw}}. \tag{3.20}$$

■ 3.8 Steering Angle Projection

The following equation maps the forces acting on the front and rear wheel, respectively, on the forces and rotational moment acting in the center of mass. The coordinate frame of the front wheel is shifted and has different rotation from the coordinate frame of the vehicle, given by the steering angle δ; whereas the coordinate frame of the rear wheel is only shifted:

$$
\begin{bmatrix} F_x \\ F_y \\ M_z \end{bmatrix} = \begin{bmatrix} \cos\delta & -\sin\delta \\ \sin\delta & \cos\delta \\ l_\mathrm{f}\sin\delta & l_\mathrm{f}\cos\delta \end{bmatrix} \cdot \begin{bmatrix} F_\mathrm{xf} \\ F_\mathrm{yf} \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & -l_\mathrm{r} \end{bmatrix} \cdot \begin{bmatrix} F_\mathrm{xr} \\ F_\mathrm{yr} \end{bmatrix} =
\tag{3.21}
$$

$$
= \begin{bmatrix} \cos\delta & -\sin\delta & 1 & 0 \\ \sin\delta & \cos\delta & 0 & 1 \\ l_\mathrm{f}\sin\delta & l_\mathrm{f}\cos\delta & 0 & -l_\mathrm{r} \end{bmatrix} \cdot \begin{bmatrix} F_\mathrm{xf} \\ F_\mathrm{yf} \\ F_\mathrm{xr} \\ F_\mathrm{yr} \end{bmatrix}.
\tag{3.22}
$$

■ 3.9 Rigid Body Dynamics

With rigid body dynamics, we also take into account the aerodynamic drag:

$$
F_{\mathrm{air},x} = \frac{1}{2} \cdot c_\mathrm{air} \cdot \rho \cdot A \cdot (v \cdot \cos\beta)^2 \,,
\tag{3.23}
$$

$$
F_{\mathrm{air},y} = \frac{1}{2} \cdot c_\mathrm{air} \cdot \rho \cdot A \cdot (v \cdot \sin\beta)^2 \,,
\tag{3.24}
$$

where c_air is the drag coefficient, ρ is the density of air, and A is the frontal area of the vehicle.

The expression for the state derivatives contains an inertia matrix, a rotational matrix employing the sideslip angle β, and a force/torque vector acting on the center of mass. The vehicle sideslip is also being changed by the yaw rate r:

$$
\begin{bmatrix} \dot\beta \\ \dot v \\ \dot r \end{bmatrix} = \begin{bmatrix} \frac{1}{mv} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{I} \end{bmatrix} \cdot \begin{bmatrix} -\sin\beta & \cos\beta & 0 \\ \cos\beta & \sin\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \left(\begin{bmatrix} F_x \\ F_y \\ M_z \end{bmatrix} - \begin{bmatrix} F_{\mathrm{air},x} \\ F_{\mathrm{air},y} \\ 0 \end{bmatrix} \right) - \begin{bmatrix} r \\ 0 \\ 0 \end{bmatrix}.
$$

$$
\tag{3.25}
$$

3.10 Model Variables and Parameters

As stated earlier, the tables with single-track model variables (3.1, 3.4) and parameters (3.2, 3.3) are located in this section.

Table 3.1: Variables used in the single-track vehicle model.

Variable	Symbol	Unit
Sideslip angle	β	rad
Velocity of the vehicle in the centre of gravity (CG)	v	m/s
Yaw rate (vehicle angular velocity)	r	rad/s
Angular velocity of the front wheel/axle	ω_f	rad/s
Angular velocity of the rear wheel/axle	ω_r	rad/s
Steering angle	δ	rad
Front wheel x-velocity	$v_{x\mathrm{f}}$	m/s
Front wheel y-velocity	$v_{y\mathrm{f}}$	m/s
Rear wheel x-velocity	$v_{x\mathrm{r}}$	m/s
Rear wheel y-velocity	$v_{y\mathrm{r}}$	m/s
Sideslip angle of the front wheel	α_f	rad
Sideslip angle of the rear wheel	α_r	rad
Slip ratio of the front wheel	λ_f	–
Slip ratio of the rear wheel	λ_r	–
Logitudinal force acting on the front wheel	$F_{x\mathrm{f}}$	N
Lateral force acting on the front wheel	$F_{y\mathrm{f}}$	N
Logitudinal force acting on the rear wheel	$F_{x\mathrm{r}}$	N
Lateral force acting on the rear wheel	$F_{y\mathrm{r}}$	N
Longitudinal force acting on the vehicle	F_x	N
Lateral force acting on the vehicle	F_y	N
Rotational torque of the vehicle	M_z	Nm
Longitudinal aerodynamic drag force	$F_{\mathrm{air},x}$	N
Lateral aerodynamic drag force	$F_{\mathrm{air},y}$	N

Table 3.2: Parameters used in the single-track vehicle model.

Parameter	Symbol	Value	Unit
Distance from the front axle to the CG	l_f	1.0228	m
Distance from the rear axle to the CG	l_r	1.3502	m
Wheel radius	p	0.2765	m
Mass of the vehicle	m	942	kg
Yaw moment of intertia	I	2500	$\mathrm{kg \cdot m^2}$
Gravitational acceleration	g	9.81	$\mathrm{m \cdot s^{-2}}$
Front axle normal force	F_zf	5,258	$\mathrm{km \cdot m \cdot s^{-2}}$
Rear axle normal force	F_zr	3,983	$\mathrm{km \cdot m \cdot s^{-2}}$
Drag coefficient	c_air	0.4	–
Density of air	ρ	1.2	$\mathrm{kg \cdot m^{-3}}$
Vehicle frontal area	A	3	$\mathrm{m^2}$

Table 3.3: Parameters for tire modeling used in the single-track vehicle model.

Parameter	Symbol	Value	Unit
Shaping coefficients for longitudinal dynamics of the front wheel	$c_{\mathrm{D},xf}$	1.4	–
	$c_{\mathrm{B},xf}$	16	–
	$c_{\mathrm{C},xf}$	1.3	–
	$c_{\mathrm{E},xf}$	-10	–
Shaping coefficients for longitudinal dynamics of the rear wheel	$c_{\mathrm{D},xr}$	0.3	–
	$c_{\mathrm{B},xr}$	100	–
	$c_{\mathrm{C},xr}$	1	–
	$c_{\mathrm{E},xr}$	-15	–
Shaping coefficients for lateral dynamics (the same for all wheels)	$c_{\mathrm{D},y}$	1.1	–
	$c_{\mathrm{B},y}$	4	–
	$c_{\mathrm{C},y}$	1.3	–
	$c_{\mathrm{E},y}$	-20	–

Table 3.4: Variables used in the force scaling algorithm in single-track vehicle model.

Variable	Symbol	Unit
Logitudinal force on front wheel (not adjusted)	$F_{xf,\text{raw}}$	N
Lateral force on front wheel (not adjusted)	$F_{yf,\text{raw}}$	N
Logitudinal force on rear wheel (not adjusted)	$F_{xr,\text{raw}}$	N
Lateral force on rear wheel (not adjusted)	$F_{yr,\text{raw}}$	N
Help variables	β_f^*	–
for the force scaling algorithm	β_r^*	rad
	$\mu_{xf,\text{act}}$	–
Friction coefficients	$\mu_{xr,\text{act}}$	–
for the force scaling algorithm	$\mu_{yf,\text{act}}$	–
	$\mu_{yr,\text{act}}$	–
	μ_{xf}	–
Friction coefficients	μ_{yf}	–
for the force scaling algorithm	μ_{xr}	–
	μ_{yr}	–

14

Chapter 4

Driving Envelope Definition

The driving envelope is a set of boundaries on selected model variables. These boundaries can be grouped into the *lateral*, *longitudinal*, or *combined slip envelope*, by the physical meaning of the variables used.

These boundary types can be applied to each wheel. The single-track model is used in this thesis, therefore the following equations set boundaries on one front wheel and one rear wheel. Taking a twin-track model, the number of inequalities for each axle will double, or the average values can be limited instead.

The boundaries placed divide the values of the selected variables into two groups: those which are allowed (safe) and those which should be avoided. The latter group contains values of the envelope variables for which the model behavior is not well defined, the vehicle is not controllable, or the vehicle state is unpredictable for the driver.

4.1   Lateral Envelope

The lateral envelope represents the boundaries on the sideslip angles:

$$\alpha_{f,min} \leq \alpha_f \leq \alpha_{f,max}, \tag{4.1}$$

$$\alpha_{r,min} \leq \alpha_r \leq \alpha_{r,max}. \tag{4.2}$$

Limits are typically chosen around the sideslip angle values for which the lateral forces given by the Pacejka Magic Formula have their extremes.

The closer the limits are to 0, the more predictable and safe the vehicle behavior is because Pacejka Magic Formula has very linear behavior around 0. On the other hand, there is also motivation to set the limits further from 0 (even further than the extremes arguments). This allows the car to slip and go with "drifty" behavior, which can be beneficial when aiming for high speeds around a sharp curve (typically in racing).

4.2 Longitudinal Envelope

The longitudinal envelope bounds the slip ratios,

$$\lambda_{\mathrm{f,min}} \leq \lambda_{\mathrm{f}} \leq \lambda_{\mathrm{f,max}}, \tag{4.3}$$

$$\lambda_{\mathrm{r,min}} \leq \lambda_{\mathrm{r}} \leq \lambda_{\mathrm{r,max}}, \tag{4.4}$$

and prevents situations in which too little road contact is maintained — wheel lockage and wheel slippage, for example — reducing the tire wear.

For the front wheel, the limits on the slip ratios are again set at the outer neighborhood points of the peaks of the Pacejka Magic Formula function.

For the car used, the rear wheel Pacejka Magic Formula plot does have its extremes only in the slip ratio extremes itself; the limits for the rear tire are set equal to the limits of the front tire merely to prevent tire wear.

4.3 Combined Slip Envelope

The combined slip envelope resembles the physical limit mentioned already in Section 3.7. The force vector generated by a tire cannot be higher than the normal force acting on the tire.

Therefore, the limit can be different for a different car, but one does not have to choose it using a guess, it is strictly given. Perhaps, a slightly different limit can be set to account for inaccuracies of the model, such as assuming constant load forces. In this work, the default limit is kept.

Unlike in the single-track model description in Chapter 3, the constraints here are stated using the *normalized Pacejka functions $P_{x\mathrm{f}}$, $P_{x\mathrm{r}}$, $P_{y\mathrm{f}}$, $P_{y\mathrm{r}}$*:

$$\sqrt{P_{x\mathrm{f}}^2(\lambda_{\mathrm{f}}) + P_{y\mathrm{f}}^2(\alpha_{\mathrm{f}})} \leq \mu, \tag{4.5}$$

$$\sqrt{P_{x\mathrm{r}}^2(\lambda_{\mathrm{r}}) + P_{y\mathrm{r}}^2(\alpha_{\mathrm{r}})} \leq \mu, \tag{4.6}$$

where

$$P_{x\mathrm{f}} = \frac{F_{x\mathrm{f}}}{F_{z\mathrm{f}} \cdot c_{\mathrm{D},x,\mathrm{f}}}, \quad P_{y\mathrm{f}} = \frac{F_{y\mathrm{f}}}{F_{z\mathrm{f}} \cdot c_{\mathrm{D},y}}, \tag{4.7}$$

$$P_{x\mathrm{r}} = \frac{F_{x\mathrm{r}}}{F_{z\mathrm{r}} \cdot c_{\mathrm{D},x,\mathrm{r}}}, \quad P_{y\mathrm{r}} = \frac{F_{y\mathrm{r}}}{F_{z\mathrm{r}} \cdot c_{\mathrm{D},y}}. \tag{4.8}$$

By the definition of the forces, it is implied: $|P_{x\mathrm{f}}| \leq 1$, $|P_{y\mathrm{f}}| \leq 1$, $|P_{x\mathrm{r}}| \leq 1$, $|P_{y\mathrm{r}}| \leq 1$. The constant μ is the tire-to-road friction coefficient.

The normalized alternative was chosen to keep the order of the slack variables introduce in Chapter 7 similar.

16

Chapter **5**

Modeling Acceleration and Braking

The references of the NMPC are the variables which are directly controlled by the driver: the state of the accelerator pedal, brake pedal, and the steering wheel. While the state of the steering wheel has its counterpart in the steering angle of the front wheel in the single-track model, the other two inputs of the single-track model are the angular velocities of the front and rear wheel. This leads to the need for a model of the pedals.

Let us denote the state of

- the accelerator pedal as $a \in [0, 1]$ and

- that of the brake pedal as $b \in [0, 1]$,

where 0 is the default state and 1 denotes a fully depressed pedal.

This chapter attempts to describe relationships which map the state of pedals (a, b) to the wheel angular velocities $(\omega_{\mathrm{f}}, \omega_{\mathrm{r}})$ which the car would have at some point in the future if the information from the pedals is passed without any changes or delay.

The fundamental source of the following formulas is the technical report [6].

▇ 5.1 **Wheel Dynamics**

As stated in [6] (and similarly in [2]), front wheel dynamics are governed by the following differential equation:

$$\dot{\omega}_{\mathrm{f}} = \frac{\tau_{\mathrm{E}} + \tau_{\mathrm{Rf}} - \tau_{\mathrm{Bf}} - \tau_{\mathrm{Df}}}{J}, \tag{5.1}$$

where J is the polar moment of inertia of a wheel, τ_{E} is the torque on the wheel from the engine, τ_{Rf} is the torque cause by the reaction force, τ_{Bf} is the braking torque, and τ_{Df} is the torque due to friction.

The car used in the simulator is front wheel drive, therefore the engine torque τ_{E} acts on the front wheel only and does not have a subscript. A dynamic equation for the rear wheel is simpler:

$$\dot{\omega}_{\mathrm{r}} = \frac{\tau_{\mathrm{Rr}} - \tau_{\mathrm{Br}} - \tau_{\mathrm{Dr}}}{J}, \tag{5.2}$$

with τ_{Rr} being the reaction torque, τ_{Br} braking torque, and τ_{Dr} friction torque on the rear wheel.

▇ 5.2 **Engine Torque**

In order to calculate the torque applied to the wheels by the engine, the vehicle powertrain is modeled. The powertrain of a car includes the engine, a torque converter, a gearbox and final drive differential. The torque converter is here neglected (assumed to be locked), as well as the servomotor dynamics of the throttle and brakes and any behavior of the reverse drive.

The car used in simulation has 5 forward gears. They are represented by five gear ratios: η_g, where $g \in \{1, 2, 3, 4, 5\}$. The final drive ratio is denoted η_{f}.

Assuming no clutch slippage and a locked torque converter, the engine rotation rate is a function of the front wheel's angular velocity:

$$\omega_{\mathrm{E}} = \begin{cases} \eta_g \cdot \eta_{\mathrm{f}} \cdot \omega_{\mathrm{f}} & \text{if } \eta_g \cdot \eta_{\mathrm{f}} \cdot \omega_{\mathrm{f}} > \omega_{\mathrm{E0}}, \\ \omega_{\mathrm{E0}} & \text{otherwise,} \end{cases} \tag{5.3}$$

where ω_{E0} is the idling speed.

The torque the engine is able to generate depends on the engine rotation rate. This relationship was measured in the simulation environment and is approximated by three second-order polynomials. The resulting engine torque curve is plotted in Figure 5.1. The maximal engine torque (at the crankshaft) $T_{\mathrm{E,max}} = 130$ Nm is at the rotation rate of 5,438 revolutions per minute (RPM).

18
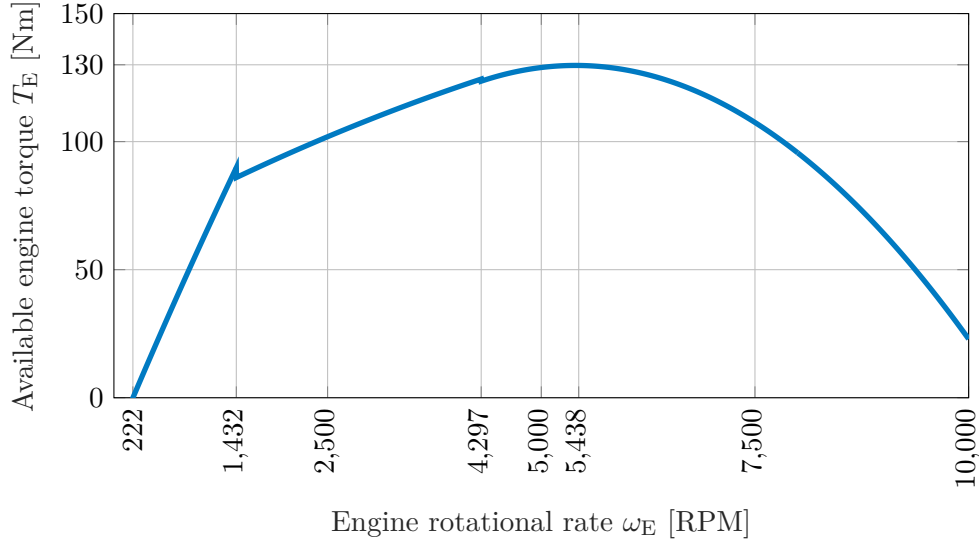
Figure 5.1: Engine torque curve

The quadratic approximation function notation follows:

$$T_{\mathrm{E}} = \kappa_{s,2} \cdot \omega_{\mathrm{E}}^2 + \kappa_{s,1} \cdot \omega_{\mathrm{E}} + \kappa_{s,0}, \tag{5.4}$$

where $s \in \{1, 2, 3\}$ denotes the interval for which is the approximation valid.

Although a car with an automatic gearbox is used, the simulator Live for Speed does provide the state of clutch. Let us denote the command signal coming from the clutch pedal in the racing simulator by $c \in [0, 1]$, where $c = 0$ denotes disengaged disks and $c = 1$ means the disks are pressed to each other with maximum available force. In a car (with manual gearbox), the clutch is placed between the engine and gearbox. It is used to temporarily separate the engine (crankshaft) from the gearbox.

The model uses a very simple formula similar to the Coulomb friction model in [7]. For more complex model, the engine dynamics would have to be introduced. The energy transfer coefficient is:

$$\mu_{\mathrm{E}} = c \cdot a \tag{5.5}$$

The torque delivered by the engine to the wheel is a product of the maximal torque available at the engine:

$$\tau_{\mathrm{E}} = \mu_{\mathrm{E}} \cdot \eta_{\mathrm{d}} \cdot \eta_g \cdot \eta_{\mathrm{f}} \cdot T_{\mathrm{E}}, \tag{5.6}$$

where η_{d} is the drivetrain efficiency.

5.3   Reaction Torque

The reaction torque is caused by the longitudinal force generated by the tire.

As the slip ratio here is defined in opposite way to the one in [6], the sign in the expression for the reaction torque is also opposite:

$$\tau_{\mathrm{Rf}} = -p \cdot F_{x\mathrm{f}}, \quad \tau_{\mathrm{Rr}} = -p \cdot F_{x\mathrm{r}}, \tag{5.7}$$

where p is the wheel rolling radius.

5.4   Braking Torque

The dynamics of the brakes are neglected. The braking torques are computed using the linear equations

$$\tau_{\mathrm{Bf}} = b \cdot \mathrm{sign}\,(\omega_{\mathrm{f}}) \cdot \xi_{\mathrm{f}} \cdot T_{\mathrm{B,max}}, \text{ and} \tag{5.8}$$

$$\tau_{\mathrm{Br}} = b \cdot \mathrm{sign}\,(\omega_{\mathrm{r}}) \cdot (1 - \xi_{\mathrm{f}}) \cdot T_{\mathrm{B,max}}, \tag{5.9}$$

where $\tau_{\mathrm{B,max}}$ is the maximal torque which can be applied on one axle by the brakes. The ratio ξ_{f} is the brake force distribution. The torque has to act against the angular velocity, therefore the presence of $\cdot\mathrm{sign}(\omega_{\mathrm{f}})$ and $\cdot\mathrm{sign}(\omega_{\mathrm{r}})$.

5.5   Friction Torque

Viscous friction is considered where k is the viscous friction coefficient:

$$\tau_{\mathrm{Df}} = k \cdot \omega_{\mathrm{f}}, \quad \tau_{\mathrm{Dr}} = k \cdot \omega_{\mathrm{r}}. \tag{5.10}$$

5.6 Variables and Parameters

Table 5.1: Variables used in the model for wheel dynamics.

Variable/Parameter	Symbol	Value	Unit
Accelerator command	a	$[0,1]$	–
Brake command	b	$[0,1]$	–
Clutch command	c	$[0,1]$	–
Front wheel angular velocity	ω_f	–	$\mathrm{rad \cdot s^{-1}}$
Rear wheel angular velocity	ω_r	–	$\mathrm{rad \cdot s^{-1}}$
Engine torque (front wheel)	τ_E	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Engine angular velocity	ω_E	–	$\mathrm{rad \cdot s^{-1}}$
Maximum available engine torque at the moment	T_E	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Energy transfer coefficient	μ_E	–	–
Reaction torque (front wheel)	τ_Rf	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Reaction torque (rear wheel)	τ_Rr	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Braking torque (front wheel)	τ_Bf	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Braking torque (rear wheel)	τ_Br	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Friction torque (front wheel)	τ_Df	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$
Friction torque (rear wheel)	τ_Dr	–	$\mathrm{kg \cdot m^2 \cdot s^{-2}}$

Table 5.2: Parameters used in the model for wheel dynamics.

Parameter	Symbol	Value	Unit
Wheel moment of inertia	J	45	$\mathrm{kg \cdot m^2}$
Density of air	ρ	1.2	$\mathrm{kg \cdot m^{-3}}$
Frontal area of the vehicle	A	3	$\mathrm{m^2}$
1st gear ratio	η_1	4.9013	–
2nd gear ratio	η_2	3.0276	–
3rd gear ratio	η_3	2.1802	–
4th gear ratio	η_4	1.7284	–
5th gear ratio	η_5	1.4000	–
Final drive ratio	η_f	3.2000	–
Drivetrain efficiency	η_d	0.8500	–
Torque curve coefficients	$\kappa_{1,2}$	$-3.1588 \cdot 10^{-4}$	–
for $\omega_\mathrm{E} \leq 150$ rad/s	$\kappa_{1,1}$	0.7640	–
$\Rightarrow s = 1$	$\kappa_{1,0}$	-17.6322	–
Torque curve coefficients	$\kappa_{2,2}$	$-7.6377 \cdot 10^{-5}$	–
for 150 rad/s $< \omega_\mathrm{E} \leq 450$ rad/s	$\kappa_{2,1}$	0.1739	–
$\Rightarrow s = 2$	$\kappa_{2,0}$	61.6104	–
Torque curve coefficients	$\kappa_{3,2}$	$-4.6032 \cdot 10^{-4}$	–
for 450 rad/s $< \omega_\mathrm{E}$	$\kappa_{3,1}$	0.5206	–
$\Rightarrow s = 3$	$\kappa_{3,0}$	-17.4539	–
Idling speed	ω_E0	99.4500	$\mathrm{rad \cdot s^{-1}}$

Chapter 6

MPC Solvers

6.1 Transcription Methods

The following summary of transcription methods is based on [8].

Control problems are typically approached with a combination of the following options:

- indirect or direct,

- shooting method or collocation method,

- an "h-method", or a "p-method".

In the real world, there are typically continuous systems, whereas controllers are discrete systems. An *indirect method* means the system is first optimized as-is (using continuous equations) and then discretized. In a *direct method*, the system is firstly discretized, and the optimization is done using a discrete system description. The latter tends to be less accurate but easier to devise and solve.

A *shooting method* is based on simulation. We further divide shooting methods into single shooting and multiple shooting methods. Using a *single shooting* method, the system state at any stage is formulated using the system variables from only the first stage, which leads to a high level of complexity in further stages. However, for highly nonlinear systems, a simulation using this method can represent the system behaviour better. In a *multiple shooting* method, there are optimization variables representing the system states for every stage. Any dependencies bounding the states are defined only between two adjacent stages.

The shooting methods are based on simulation, while *collocation methods* are based on function approximations. For example, instead of integrating the differential equations of a system, an approximation can be made with a polynomial function. The system state values are forced to be precise only at certain points. Unlike shooting metods, collocation methods are better suited

for problems with more complicated constraints.

An *h-method* is a low-order method which converges by increasing the number of segments, whereas *p-methods* converge by increasing the method order.

■ 6.2 Available Software

The software originally assigned for this thesis, CasADi [9], is an open-source software tool for gradient-based numerical optimization. CasADi provides great flexibility for formulation of various optimization tasks, a high level of comfort for specifying optimal control problems, and an easy approach to debugging with its Opti class. However, CasADi is not an optimal control problem (OCP) solver.

By default, CasADi uses the open-source solver Ipopt [10]. It is said to be a robust code, or even "the state-of-the-art interior-point solver" [11]. The solver is shipped together with CasADi, and uses the open-source linear solver MUMPS [12] by default.

However, Ipopt recommends using a linear solver other than MUMPS. Using the MA57, MA27, MA77 and similar linear solvers form the HSL library [13], it was possible to speed up the solving times by approximately one third compared to MUMPS (given the relatively small problem size).

The solver SNOPT [14] is said to be the industry standard for nonlinear optimization. Although a free academic license is available, this commercial solver cannot be distributed with CasADi. The last CasADi version seems to be incompatible with SNOPT (tried with the MATLAB version of CasADi 3.5.5 and precompiled SNOPT 7.7.5 on Windows).

■ 6.3 Selected Software

The solver finally used is a specifically generated software produced by the commercial software FORCESPRO by Embotech [11, 15, 16]. The company provided a time-limited free academic license and support.

For a nonlinear optimization problem, FORCESPRO generates an interior-point method based solver using the direct multiple shooting method [15]. They claim their solvers are comparable or faster than Ipopt, which was found to be true in case of the NMPC formulation used in this thesis, with multiple-shooting transcription in Opti using the MA57 linear solver.

Chapter 7

MPC Formulation

Using the FORCESPRO high-level interface, it is possible to generate a variant of their FORCES NLP solver to solve a non-convex finite-time nonlinear optimization control problem in a certain form. This form is stated in the FORCESPRO user manual [16].

For better correspondence with the source code defining the solver, the formulation of this particular NMPC task is described using the same notation.

7.1 Differences from Previously Defined Models

The MPC formulation uses the results of previous chapters, if not stated otherwise. This section lists the most important differences.

7.1.1 Raw Reaction Forces

The algorithm from Section 3.7 (Equations 3.14 to 3.19) is not easily approximated as a continuously differentiable function, therefore it is not used in the solver. As reaction forces, their raw values are taken:

$$F_{x\mathrm{f}} = F_{x\mathrm{f,raw}}, \quad F_{y\mathrm{f}} = F_{y\mathrm{f,raw}}, \quad F_{x\mathrm{r}} = F_{x\mathrm{r,raw}}, \quad F_{y\mathrm{r}} = F_{y\mathrm{r,raw}}. \tag{7.1}$$

7.1.2 No Constraint on Combined Slip

Even though the boundaries expressed in Section 4.3, Equation 4.5, resembling the motivation for the algorithm mentioned in the previous paragraph, are implementable in the solver, they are also not used.

No weights for which would this constraint be beneficial were found. A reason could be that the function is not trivial, employing a square root of an expression of the reaction forces, which are represented by nontrivial functions themselves. This aspect has to be further investigated.

◼ 7.1.3 Sideslip Angles Simplification

An absolute value is not a continuously differentiable function, and so it can cause problems when using gradient-based optimization solvers. As the experiments are done in a racing simulator, we assume only forward driving, and the solver is called only when the vehicle velocity is ≥ 1 m/s. The longitudinal velocities in the denominators are then assumed to be always positive and the absolute values can therefore be omitted:

$$\alpha_{\mathrm{f}} = -\arctan\frac{v_{y\mathrm{f}}}{v_{x\mathrm{f}}}, \quad \alpha_{\mathrm{r}} = -\arctan\frac{v_{y\mathrm{r}}}{v_{x\mathrm{r}}}. \tag{7.2}$$

◼ 7.1.4 Slip Ratios Approximation

For the same reason, the absolute values are also omitted in the function definitions for the longitudinal slip rations.

Besides the absolute values, the original Equations 3.3 contain a maximal value. This function was not omitted, because it limits the resulting slip ratio in the interval $[-1, 1]$. Values far from this interval would yield in a disproportionate cost, making it hard or impossible to find suitable weighting factors.

The maximal value is approximated by a smooth function instead, as suggested in [17]:

$$\lambda_{\mathrm{f}} = \frac{\omega_{\mathrm{f}}p - v_{x\mathrm{f}}}{\frac{1}{2} \cdot \left(\omega_{\mathrm{f}}p + v_{x\mathrm{f}} + \sqrt{(\omega_{\mathrm{f}}p - v_{x\mathrm{f}})^2 + \varepsilon^2} \right)},$$

$$\lambda_{\mathrm{r}} = \frac{\omega_{\mathrm{r}}p - v_{x\mathrm{r}}}{\frac{1}{2} \cdot \left(\omega_{\mathrm{r}}p + v_{x\mathrm{r}} + \sqrt{(\omega_{\mathrm{r}}p - v_{x\mathrm{r}})^2 + \varepsilon^2} \right)},$$

where ε is a small constant, here $\varepsilon = 10^{-4}$.

◼ 7.2 Minimization Task

The primary goal is to minimize a sum of values of the objective function,

$$\sum_{k=1}^{N-1} f_k(\boldsymbol{z}_k, \boldsymbol{p}_k), \tag{7.3}$$

where N is the length of the prediction horizon (the number of stages), k is the index of a stage, $\boldsymbol{z}_k \in \mathbb{R}^{n_k}$ is the vector of the optimization variables in stage k, $\boldsymbol{p}_k \in \mathbb{R}^{l_k}$ is the vector of runtime parameters for stage k, and $f_k : \mathbb{R}^{n_k} \times \mathbb{R}^{l_k} \to \mathbb{R}$ is the objective/cost function for stage k.

The length of the prediction horizon was chosen to be $N = 10$. It is equal to the prediction horizon of the linear MPC in [1], and also empirically proved

to work well. Together with the timestep $T_\mathrm{s} = 20$ ms, the prediction horizon is timewise 180 ms long.

Notation. All optimization variables are defined by stage and to be precise, they should be written with index k. However, this is not necessary, because the stages do not differ in this aspect. Where is this index missing, it was likely omitted for simplicity.

▊ 7.3 Optimization Variables

Multiple variants of the NMPC formulation were tried. The final one consists of seventeen optimization variables in each stage. Besides the basic ones for model states and inputs, there are variables for approximation of absolute values, one to build a slew of rate constraint, and some used as slacks to create soft constraints.

All NMPC variables are listed in Table 7.1, together with their units. In the order used in the source code, there are

- 5 slack variables for constraints: s_δ, s_{α_f}, s_{α_r}, s_{λ_f}, s_{λ_r};

- 3 variables representing the absolute values of the deviation from the references:

  - d_a, for the value $|a_\mathrm{ref} - a|$,
  - d_b, for the value $|b_\mathrm{ref} - b|$,
  - d_δ, for the value $|\delta_\mathrm{ref} - \delta|$;

- 3 control input variables: a, b, δ,

- 1 variable to store the control input value computed in the previous stage, as it is only possible to combine the variables of one single stage in FORCESPRO: $\delta'_k = \delta_{k-1}$;

- 2 variables representing the state of the wheels: ω_f, ω_r;

- 3 variables representing the state of the vehicle as defined in the single-track model: β, v, r.

The vector $\boldsymbol{z}_k$ is then

$$\boldsymbol{z}_k = [s_{\delta,k} \quad s_{\alpha_\mathrm{f},k} \quad s_{\alpha_\mathrm{r},k} \quad s_{\lambda_\mathrm{f},k} \quad s_{\lambda_\mathrm{r},k} \quad d_{a,k} \quad d_{b,k} \quad d_{\delta,k}$$
$$a_k \quad b_k \quad \delta_k \quad \delta'_k \quad \omega_{\mathrm{f},k} \quad \omega_{\mathrm{r},k} \quad \beta_k \quad v_k \quad r_k]^\top, \quad (7.4)$$

its length — seventeen elements — is the same for all stages, from $k = 1$ to $k = 10$.

27

Table 7.1: Summary of NMPC optimization variales

Variable	Symbol	Unit
Slack for slew rate constraint on δ	s_δ	rad/s^2
Slack for envelope protection constraint on α_f	s_{α_f}	rad
Slack for envelope protection constraint on α_r	s_{α_r}	rad
Slack for envelope protection constraint on λ_f	s_{λ_f}	–
Slack for envelope protection constraint on λ_r	s_{λ_r}	–
Deviation of the accelerator pedal position from ref.	d_a	–
Deviation of the brake pedal position from reference	d_b	–
Deviation of the steering angle from reference	d_δ	rad
Accelerator pedal position (NMPC output)	a	–
Brake pedal position (NMPC output)	b	–
Steering angle (NMPC output)	δ	rad
Previous value of the steering angle (previous NMPC output or δ of the preceding stage)	δ'	rad
Front wheel angular velocity	ω_f	rad/s
Rear wheel angular velocity	ω_r	rad/s
Slip angle	β	rad
Vehicle velocity	v	m/s
Yaw rate	r	rad/s

For the variables d_a, d_b, d_δ, the absolute value approximation is taken in a different manner than the one of maximal value described in Section 7.1.4. As these variables are present in the cost function which is **minimized**, it is sufficient to set lower bounds for them in the optimization problem constraints.

For example, let us consider d_a. This variable has to represent the value of $|a_\mathrm{ref} - a|$. The lower bounds are then set to $a_\mathrm{ref} - a$ and $-(a_\mathrm{ref} - a)$. As the solver minimizes the variable d_a, it becomes close to $a_\mathrm{ref} - a$ when it is positive and close to $-(a_\mathrm{ref} - a)$ when it is positive, that means, close to $|a_\mathrm{ref} - a|$. For more theoretical explanation, please refer to the term *epigraph* in [18].

7.4 Runtime Parameters

The vector of runtime parameters, representing the real-time data (those not available in the time of the solver compilation), consists of five constants which are independent on the stage:

$$\boldsymbol{p}_k = \begin{bmatrix} a_\mathrm{ref} & b_\mathrm{ref} & \delta_\mathrm{ref} & T_\mathrm{max} & \alpha_\mathrm{f,max} \end{bmatrix}^\top. \tag{7.5}$$

The runtime parameters are listed in Table 7.2. There are the three references, a parameter describing the maximum engine torque which can be applied on

the front wheel at the moment, and a velocity-dependent limit on the front sideslip angle.

Table 7.2: Summary of NMPC runtime parameters

Runtime Parameter	Symbol	Unit
Accelerator pedal position (NMPC input)	a_ref	–
Brake pedal position (NMPC input)	b_ref	–
Steering angle (NMPC input)	δ_ref	rad
Maximal available engine torque for the front wheel	T_max	Nm
Limit the sideslip angle of the front wheel	$\alpha_\text{f,max}$	rad

Computing $\alpha_\text{f,max}$. A low constant limit would unreasonably increase the minimal cornering radius of the vehicle for lower velocities, e.g., when parking. A high constant limit would not be sufficient to prevent over-slipping for high velocities.

Therefore, the current limit on the front sideslip angle is determined using the vehicle velocity (before calling the solver):

$$\alpha_\text{f,max} = \begin{cases} 0.8 & \text{if } v < 5 \text{ m/s,} \\ \frac{0.35-0.8}{20-5} \cdot v + 0.8 - \frac{0.35-0.8}{20-5} \cdot 5 & \text{if } v < 20 \text{ m/s,} \\ 0.35 & \text{otherwise.} \end{cases} \quad (7.6)$$

▊ 7.5 Cost Function

The same holds for the cost function, it is defined identically for all considered stages, from 1 to $(N-1)$:

$$\begin{aligned} f_k(\boldsymbol{z}_k, \boldsymbol{p}_k) = {}& 5 \cdot 10^3 \cdot s_{\delta,k}^2 + \\ & + 10^4 \cdot s_{\alpha_\text{f},k}^2 + 10^4 \cdot s_{\alpha_\text{r},k}^2 + \\ & + 2 \cdot 10^2 \cdot s_{\lambda_\text{f},k}^2 + 2 \cdot 10^2 \cdot s_{\lambda_\text{r},k}^2 + \\ & + \frac{1}{(a_\text{ref}+0.1)^2} \cdot d_{a,k} + \frac{10}{(b_\text{ref}+0.1)^2} \cdot d_{b,k} + \\ & + 200 \cdot d_{\delta,k} + 500 \cdot d_{\delta,k}^2 + \\ & + \beta_k^2 + r_k^2. \end{aligned} \quad (7.7)$$

The vehicle states β, r are lightly weighted to prevent unreasonable shooting of these states and to prevent spinning of the vehicle.

It is recommended to select weights for slack variables significantly higher than for any other variable. This is clearly followed for the slacks on sideslip angles.

The terms for following the references on accelerator and brake pedals positions as well as the term for following the steering wheel reference use variables which represent the absolute values of corresponding deviations. An absolute value (opposed to the square) is selected to also count for low deviations. The steering angle term employing an absolute value is additionally combined with a term employing the square.

The weights of the pedal position deviations are divided by their reference in order to avoid producing nonzero values on both pedals at the same time. (A zero reference means increasing the weight a hundred times, whereas a unitary reference divides the weight in the nominator by 1.21.)

▊ 7.6 Initial Equality

Some of the optimization variables in the first stage are replaced by a fixed value:

- the previously computed steering angle (eventually, when previous run of the solver crashed or when starting the simulation, the reference from the driver);

- and the current state of the vehicle received from the vehicle dynamics simulator (LFS).

Formulas for computing the wheel angular velocities, the vehicle velocity and the vehicle sideslip angle from values received from the vehicle dynamics simulator are in Section 8.2.

In the notation of FORCESPRO,

$$\boldsymbol{z}_1(\mathcal{I}) = \boldsymbol{z}_{\text{init}}, \tag{7.8}$$
$$\text{where } \mathcal{I} = \{12, 13, 14, 15, 16, 17\} \tag{7.9}$$
$$\text{and } \boldsymbol{z}_{\text{init}} = \begin{bmatrix} \delta_{\text{prev}} & \omega_{\text{f,LFS}} & \omega_{\text{r,LFS}} & \beta_{\text{LFS}} & v_{\text{LFS}} & r_{\text{LFS}} \end{bmatrix}^{\top}. \tag{7.10}$$

This notation efficiently translates into equalities

$$\delta_1' = \delta_{\text{prev}}, \tag{7.11}$$
$$\omega_{\text{f,1}} = \omega_{\text{f,LFS}}, \tag{7.12}$$
$$\omega_{\text{r,1}} = \omega_{\text{r,LFS}}, \tag{7.13}$$
$$\beta_1 = \beta_{\text{LFS}}, \tag{7.14}$$
$$v_1 = v_{\text{LFS}}, \tag{7.15}$$
$$r_1 = r_{\text{LFS}}, \tag{7.16}$$

where variables with the 'LFS' subscript are taken from the vehicle dynamics simulator.

7.7 Inter-stage Equality

The inter-stage equality contraint is used

- to define the relationship $\delta'_{k+1} = \delta_k$,

- and to enforce the dynamics of the states.

The dynamics are given by the differential equations implemented as MAT-LAB functions, namely the equations 5.1 and 5.2 for the wheel angular velocites, and the matrix equation 3.25 for the single-track vehicle states. The functions representing the right-hand sides of corresponding differential equations are taken as an argument of a FORCESPRO integrator function.

Although it is noted in [19] that vehicle dynamics can be relatively stiff (unstable when solving numerically) and suggests using an implicit integration scheme, it was found out that the solver is able to converge fast when a simple 1st order explicit Euler integration scheme is used.

7.8 Final equality

The final equality is unused, there are no final conditions.

7.9 Static Parameters Selection

More imporant constant limits were given a notation and are listed in Table 7.3. Those "static" parameters are used for the upper-lower bounds on optimization variables and also for the nonlinear constraints.

Table 7.3: Summary of NMPC static parameters

Variable	Symbol	Value	Unit
Time step (sampling time)	T_{s}	0.02	s
Maximal steering angle	δ_{max}	+0.5236	rad
Minimal steering angle	δ_{min}	−0.5236	rad
Maximal wheel angular velocity	ω_{max}	180	rad/s
Maximal rear sideslip angle	$\alpha_{\text{r,max}}$	+0.20	rad
Minimal rear sideslip angle	$\alpha_{\text{r,min}}$	−0.20	rad
Maximal slip ratio	λ_{max}	+0.15	–
Minimal slip ratio	λ_{min}	−0.15	–
Maximal slew of δ	δ_{slew}	0.0419	rad/s

Constants for the upper-lower bounds. The minimal and maximal steering angles ($\delta_{\min}$, $\delta_{\max}$) are given by the maximal lock of the default XF GTI car in the simulator (see Chapter 8) which is equal to $30°$. The maximal wheel angular velocity ($\omega_{\max}$) is connected to the maximal velocity the car is able to develop in the simulator.

Constants for the nonlinear constraints. The limits on the rear sideslip angle ($\alpha_{\mathrm{r,min}}$, $\alpha_{\mathrm{r,max}}$), in addition to the limits on both slip ratios ($\lambda_{\min}$, $\lambda_{\max}$), are set to be in the outer neighborhood of the force/sideslip angle and force/slip ratio characteristics (discussed in Chapter 4).

The maximal slew of δ is the limit on the change in δ between two stages. In [20], the value selected is given by the characteristics of the actuator they use in their vehicle. Although the vehicle dynamics simulator used here does not have this internal restriction, the slew is limited not to disturb the driver. In this thesis, a similar value was chosen: $120°$ per second. Taking the time step $T_{\mathrm{s}} = 0.02$ s, it leads to the value $\delta_{\mathrm{slew}} = (0.02 \cdot 120)°/\mathrm{s} = 0.0419$ rad/s.

■ 7.10 Upper-lower Bounds

Upper-lower bounds are acting directly on the optimization variables:

$$\underline{z}_k \leq z_k \leq \bar{z}_k, \tag{7.17}$$

where $\underline{z}_k$ and $\bar{z}_k$ are constant vectors.

These bounds cannot be slacked, therefore an initial guess of variables outside of the bounds can lead to an infeasibile optimization problem. The bounds should not be too strict, in fact, most of them can be filled with infinite values. However, the solver could work better with finite values.

Slack variables and the variables substituting the absolute values have to be positive. Very low but **nonzero** lower bounds for those variables were set to improve the solver performance.

For the rest of the optimization variables, the knowledge about their minimal and maximal possible values was used, and somewhat extended.

Resulting bounds are:

$$\underline{z}_k = \begin{bmatrix} 10^{-8} \\ 10^{-8} \\ 10^{-8} \\ 10^{-8} \\ 10^{-8} \\ 10^{-6} \\ 10^{-6} \\ 10^{-6} \\ 0 \\ 0 \\ \delta_{\min} \\ \delta_{\min} \\ -10 \\ -10 \\ -\pi \\ -100 \\ -15 \end{bmatrix} \le z_k = \begin{bmatrix} s_\delta \\ s_{\alpha_\mathrm{f}} \\ s_{\alpha_\mathrm{r}} \\ s_{\lambda_\mathrm{f}} \\ s_{\lambda_\mathrm{r}} \\ d_a \\ d_b \\ d_\delta \\ a \\ b \\ \delta \\ \delta' \\ \omega_\mathrm{f} \\ \omega_\mathrm{r} \\ \beta \\ v \\ r \end{bmatrix} \le \bar{z}_k = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 100 \\ 100 \\ 1.1 \\ 1.1 \\ 1.1 \\ 1 \\ 1 \\ \delta_{\max} \\ \delta_{\max} \\ \omega_{\max} \\ \omega_{\max} \\ \pi \\ 100 \\ 15 \end{bmatrix}. \tag{7.18}$$

▌ 7.11 Nonlinear Constraints

The slack variables are defined for the purpose of making some of the constraints soft, that is, to allow prospective breach of the limits they set. This is beneficial when the vehicle gets out of the driving envelope, not able to return back immediately, and to increase the solver robustness.

There are sixteen nonlinear inequalities which are equal to eight slacked constraints. Of this eight slacked constraints, four serve as soft constraints on the driving envelope, one serves as the steering angle slew protection, and three are used for the absolute value approximation.

The boundaries on the front wheel sideslip angle are set using a runtime parameter, therefore the inequality format is slightly different,

$$-10 \le -s_{\alpha_\mathrm{f},k} + \alpha_\mathrm{f} - \alpha_{\mathrm{f},\max} \le 0, \tag{7.19}$$
$$0 \le +s_{\alpha_\mathrm{f},k} + \alpha_\mathrm{f} + \alpha_{\mathrm{f},\max} \le 10, \tag{7.20}$$

from inequalities implementing the boundaries on other driving envelope variables:

$$-10 \le -s_{\alpha_\mathrm{r},k} + \alpha_\mathrm{r} \le \alpha_{\mathrm{r},\max}, \tag{7.21}$$
$$\alpha_{\mathrm{r},\min} \le +s_{\alpha_\mathrm{r},k} + \alpha_\mathrm{r} \le 10, \tag{7.22}$$
$$-100 \le -s_{\lambda_\mathrm{f},k} + \lambda_\mathrm{f} \le \lambda_{\max}, \tag{7.23}$$
$$\lambda_{\min} \le +s_{\lambda_\mathrm{f},k} + \lambda_\mathrm{f} \le 100, \tag{7.24}$$
$$-100 \le -s_{\lambda_\mathrm{r},k} + \lambda_\mathrm{r} \le \lambda_{\max}, \tag{7.25}$$
$$\lambda_{\min} \le +s_{\lambda_\mathrm{r},k} + \lambda_\mathrm{r} \le 100. \tag{7.26}$$

The slew protection:

$$-100 \leq -s_{\delta,k} + \delta \leq \delta_{\mathrm{slew}}, \tag{7.27}$$

$$-\delta_{\mathrm{slew}} \leq +s_{\delta,k} + \delta \leq 100. \tag{7.28}$$

The deviations from references:

$$10^{-6} \leq d_{a,k} - a_{\mathrm{ref}} + a_k \leq 10, \tag{7.29}$$

$$10^{-6} \leq d_{a,k} + a_{\mathrm{ref}} - a_k \leq 10, \tag{7.30}$$

$$10^{-6} \leq d_{b,k} - b_{\mathrm{ref}} + b_k \leq 10, \tag{7.31}$$

$$10^{-6} \leq d_{b,k} + b_{\mathrm{ref}} - b_k \leq 10, \tag{7.32}$$

$$10^{-6} \leq d_{\delta,k} - \delta_{\mathrm{ref}} + \delta_k \leq 10, \tag{7.33}$$

$$10^{-6} \leq d_{\delta,k} + \delta_{\mathrm{ref}} - \delta_k \leq 10. \tag{7.34}$$

▌ **7.12 Other Settings**

A time limit of 15 ms is set for the solver to provide a solution.

The linear solver selected is `symm_indefinite`, which should be a more robust alternative to the default `normal_eqs`. It solves the Karush-Kuhn-Tucker system in augmented / symmetric indefinite form, using block-indefinite factorizations; opposed to the normal equations form [16].

The CasADi version 3.5.1 is used as a tool for automatic differentiation.

When possible, the solver is provided with an initial guess from its previous output (shifted by one stage).

Chapter 8

Simulation Environment

This chapter provides a quick overview of the testing simulation environment. The environment was developed in [21], which offers more detailed description.

Following paragraphs explain the dataflow depicted in Figure 8.1.
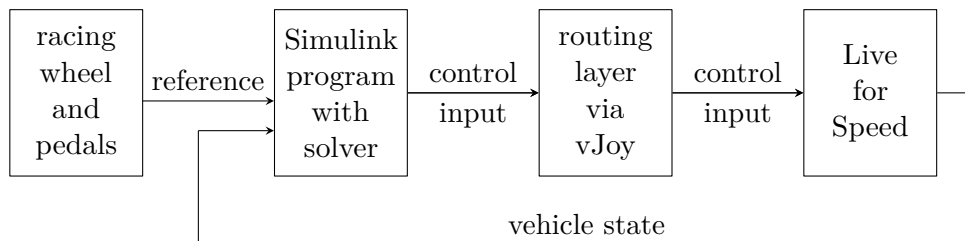
Figure 8.1: Diagram of the experimental setup

Hardware.   The system was developed on a laptop with the Intel Core i7-10750H processor (6 cores, 5 GHz maximum frequency) and 16 GB of physical memory (RAM), running on the Windows 10 operating system.

User Input.   In this particular setup, a Thrustmaster racing wheel[1] with physical pedals without a force feedback was used, but any game controller including gamepads should have similar performance.

Data Processing.   MATLAB[2] (version R2020b) serves as programming environment. In a Simulink[3] model, an input block receives the data from the game controllers and data from the simulator (below). Simulink calls the optimal control solver generated by FORCESPRO[4] and broadcasts its solution using network communication.

---

[1]`http://www.thrustmaster.com/en_US/products/ferrari-racing-wheel-red-legend-edition`
[2]`https://ch.mathworks.com/products/matlab.html`
[3]`https://ch.mathworks.com/products/simulink.html`
[4]`https://www.embotech.com/products/forcespro/overview/`

Data Forwarding. The communication from Simulink is processed by a helper program from [21] and redirected to vJoy[5] which creates a virtual game controller.

Simulator. The virtual game controller serves as an input for the Live for Speed[6] (LFS), version 0.6U11, which is a computer racecar simulator (a computer game with realistic car physics). For part of the control settings, see Figure 8.2. The simulator broadcasts the vehicle state using an UDP communication, being an input for the Simulink model.

Figure 8.2: Screenshot of settings for the virtual game controller in LFS

8.1 Selected Car

The vehicle selected for the simulation is the XF GTI with default parameters, see Figure 8.3.

The XF GTI is Live for Speed's default car. It is a front wheel drive, statically understeering car with automated transmission. It is targeted for the beginners on racing platforms.

Its parameters are listed in Section 3, Table 3.2 and 3.3. Some other parameters, relevant for wheel dynamics modeling, are also listed in Table 5.2. Most of the parameters are provided with the game, some were determined experimentally (in the simulation).

---

[5]http://vjoystick.sourceforge.net/site/
[6]https://www.lfs.net/

Figure 8.3: Screenshot of a "garage" with the XF GTI car in LFS

▌ 8.2 Vehicle State Transformation

The single-track model system states are the sideslip angle β, the translational velocity v, and the yaw rate r. (All the states are scalars.) However, the Live for Speed outputs in the same format only the yaw rate.

The sideslip angle is therefore computed using velocities v_x, v_y:

$$\beta_{\text{LFS}} = \begin{cases} -\arctan\left(\frac{v_y}{v_x}\right) & \text{if } |v_x| \geq 5, \\ 0 & \text{otherwise,} \end{cases} \tag{8.1}$$

moreover, $\beta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$.

The velocity v is computed as the size of the velocity vector represented by its elements v_x, v_y:

$$v_{\text{LFS}} = \sqrt{v_x^2 + v_y^2}. \tag{8.2}$$

As LFS simulates a twin-track vehicle, the front and rear wheel angular velocities are the arithmetical average:

$$\omega_{\text{f,LFS}} = \frac{\omega_{\text{f,left}} + \omega_{\text{f,right}}}{2}, \tag{8.3}$$

$$\omega_{\text{r,LFS}} = \frac{\omega_{\text{r,left}} + \omega_{\text{r,right}}}{2}. \tag{8.4}$$

Chapter 9

Simulation Experiments

The experiments aim to show differences of the vehicle behavior with different control strategies:

- using the developed nonlinear MPC;

- using the previously existing linear MPC;

- using exact reference from the human driver;

and possibly to prove benefits of the nonlinear MPC.

The linear MPC strategy used for the comparison is formulated in [1]. It was set to resemble the prediction horizon ($N = 10$) and timestep ($T_s = 0.02$ s) of the nonlinear MPC. Its boundaries for the front sideslip angle are ± 0.6. Besides the MPC solver, there are proportional–integral–derivative controllers computing the setpoints for the pedal position from the wheel angular valocities which are outputs of the linear MPC solver.

In the experiments, a driver (the brother of the author of this thesis) was asked to drive a selected track with each control strategy ten times (ten laps, without a break between them). From each ride, three basic numbers are noted:

- best lap time,

- mean lap time,

- and mean number of contacts with obstacles (bumps).

The track selected is *South City, Town Course*, which was chosen because it contains sharpe curves. However, this track is not flat, which can cause additional inaccuracies due to the assumtion of constant load forces.

The results, shown in Table 9.1, do not provide a clear conclusion. Timewise, the nonlinear MPC appears better than no control strategy. To prove this possible benefit, however, experiments with more drivers would have to be carried out.

Table 9.1: Ride test results.

Control Strategy	Mean Time [s]	Min. Time [s]	Bumps [−]
No control	116.74	114.15	5
Linear MPC	117.88	113.63	6
Nonlinear MPC	116.23	113.71	5

It was observed that the fatigue, motivation, and experience of the driver have significant influence on the results. Besides these factors, each driver first has to adopt to the behaviour of a vehicle with control strategy.

Subjectively, the vehicle dynamics simulator without any control strategy gave the impression of smoother (not disturbing) but slower ride than when employing on of the controllers.

The linear MPC strategy influences the pedal positions more often than the nonlinear MPC strategy, whereas the nonlinear MPC relies more on influencing the steering wheel position.

The following sections investigate the differences on the data recorded, in three distinct situations:

- cornering,

- acclererating,

- and braking.

9.1 Cornering

The investigated curve is located in the *South City, Town Course* track and is delimited by the lap distance: from 1200 to 1340 meters.
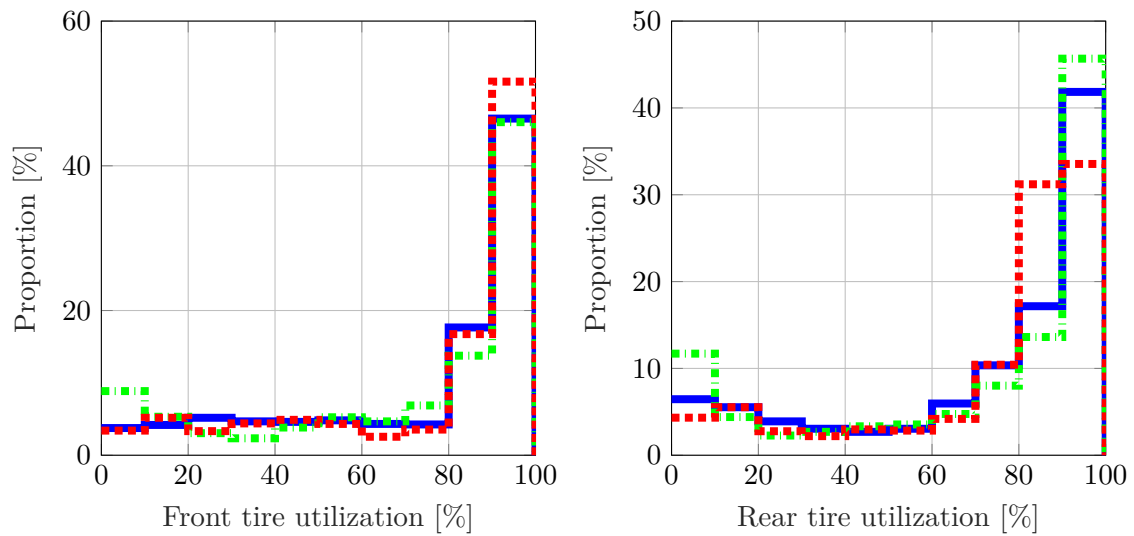
The tire utilization is defined as

$$\left| \frac{F_{y\mathrm{f}}}{c_{\mathrm{D},y} \cdot F_{z\mathrm{f}}} \right| \tag{9.1}$$

for the front tire and as

$$\left| \frac{F_{y\mathrm{r}}}{c_{\mathrm{D},y} \cdot F_{z\mathrm{r}}} \right| \tag{9.2}$$

for the rear tire. It is a dimensionless quantity which can be used as a generalization for the sideslip angles.

Higher tire utilization is correlated with higher sideslip angle. As a high sideslip angle generally leads to higher force generated on the tire, and is

Figure 9.1: Tire utilization when cornering

therefore desirable (unless the traction is lost, which would manifest the number of bumps). See Figures 9.1 and 9.2.

All the histograms, except for the ones for acceleration behavior, are computed from mean values over the ten laps. The plots of the sideslip angles (Figure 9.3, Figure 9.4) show the data of a single lap with the minimal lap time.

Figure 9.2 shows an improvement of the nonlinear MPC for the front tire utilization.

■ 9.2 Accelerating

The acceleration data show the first 70 m of the first lap. See Figures 9.5 and 9.6.

■ 9.3 Braking

As a section of the track where is common braking, was chosen a part before a curve with higher radius but with common high speed, from 2,550 m to 2,650 m. The corresponding histograms are shown on Figures 9.7 and 9.8.

The tire utilization appears to be improved, however, the distribution of the slip ratios is unchanged from the data with no control strategy.
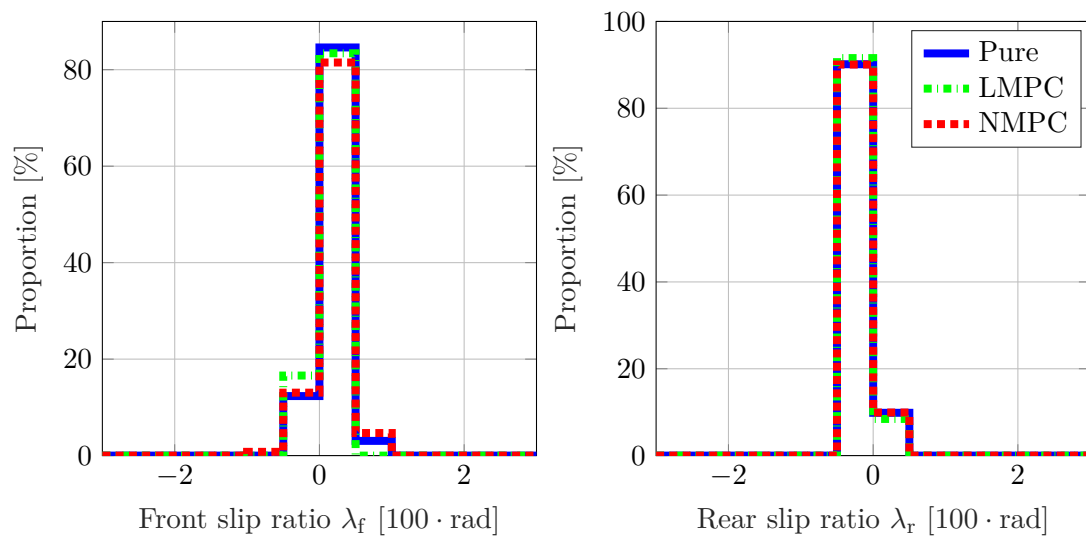
Figure 9.2: Slip ratios when cornering (the no control strategy, linear MPC, and nonlinear MPC are shown in blue, green, and red, respectivelly)
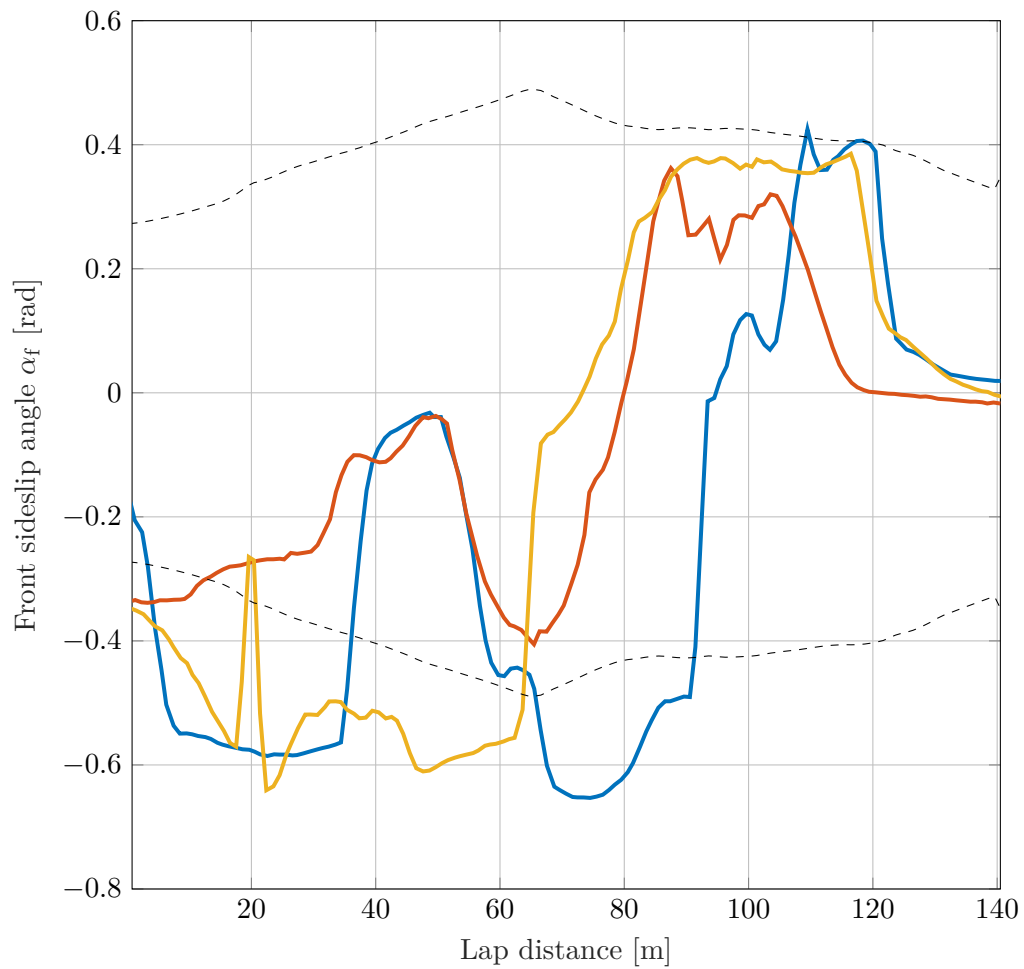
Figure 9.3: The front wheel sideslip angle when cornering. No control strategy is in blue, red the linear MPC, yellow the nonlinear MPC. The dashed line represents the velocity-dependent boundaries on the fron sideslip angle in the nonlinear MPC
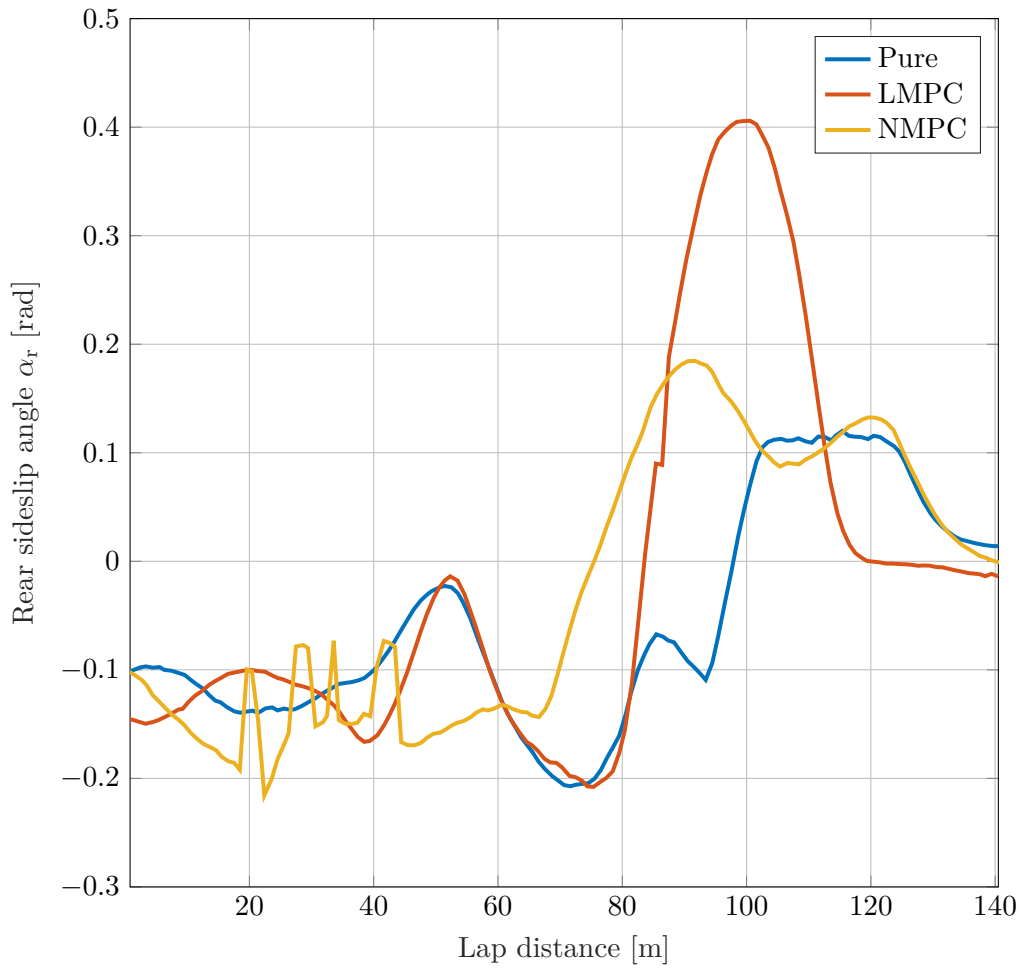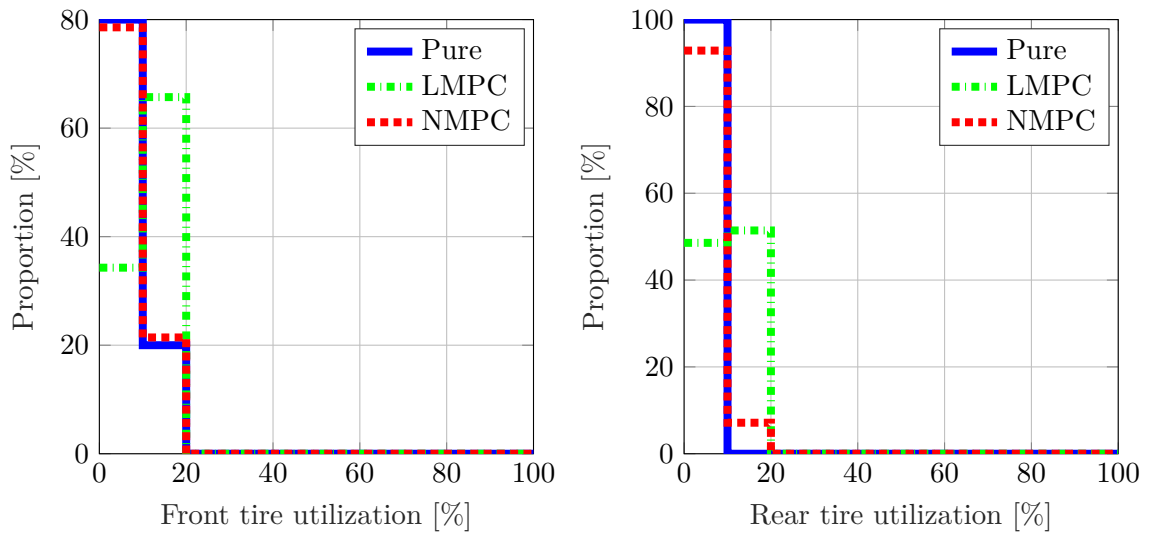
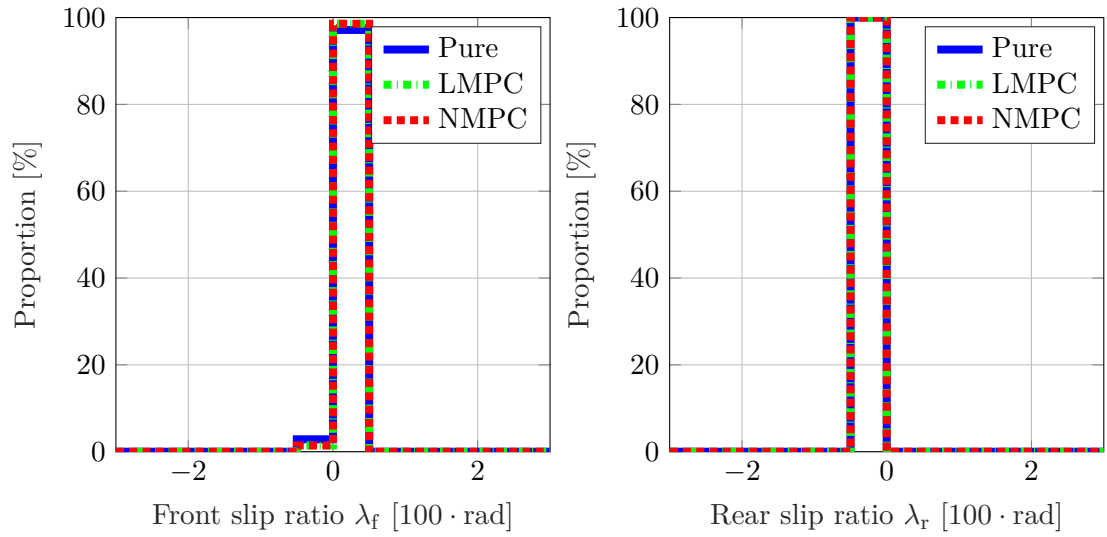**Figure 9.4:** The rear wheel sideslip angle when cornering
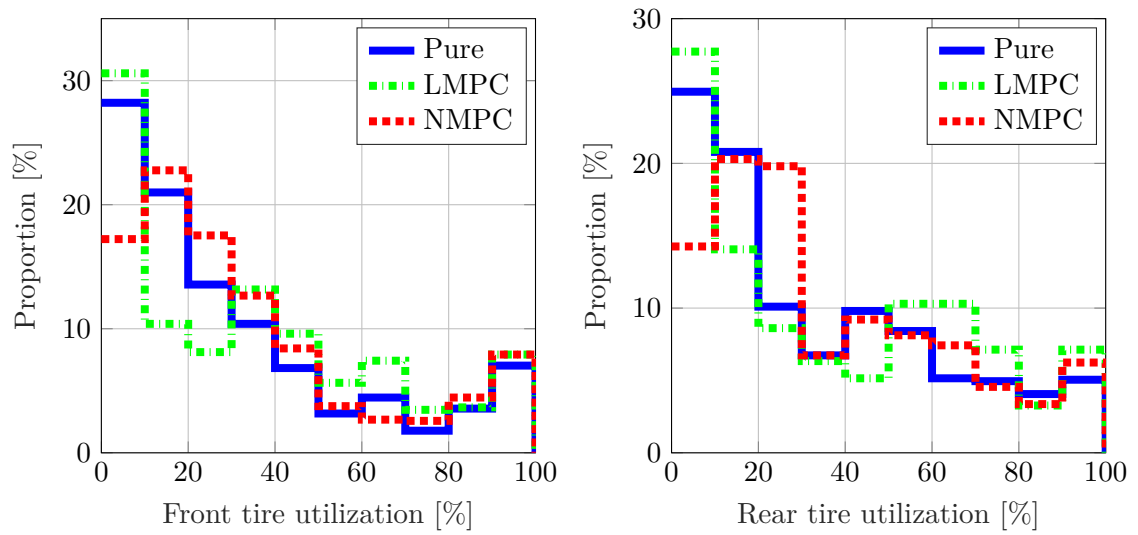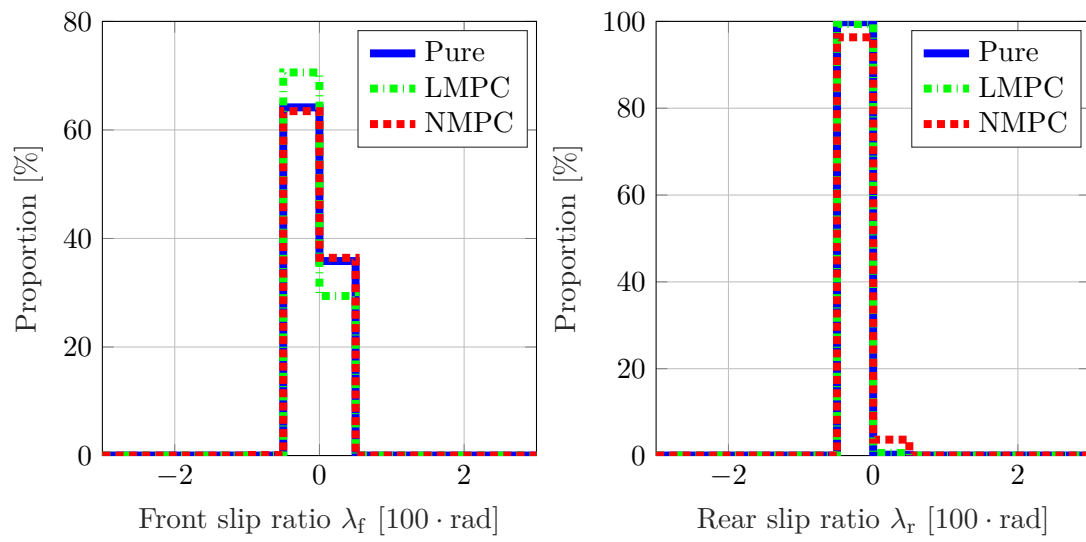

Figure 9.5: Tire utilization when accelerating

43

Figure 9.6: Slip ratios when accelerating

Figure 9.7: Tire utilization when braking

Figure 9.8: Slip ratios when braking

Chapter 10

Results

In this work,

- the *driving envelope is employed as it is defined* in [1], except for omitting the combined force restrictions for their high computational complexity and little effect, and except for loosening the limits on α_f for low velocities.

- The assigned numerical optimization *tool CasADi was not used directly*, as the *FORCESPRO solution was selected* to create the final solver. CasADi is actually used by FORCESPRO itself as an automatic differentiation tool, but FORCESPRO's architecture does rather motivate to use its own interface, so the user does not use CasADi's syntax. CasADi was used in earlier development for debugging purposes and trying other nonliear problem solvers such as Ipopt, though.

- The *single-track model was adopted* into the controller, creating approximations for some of its variables.

- A *nonlinear model predictive controller was formulated* which runs in real time.

- The *controller was tested with the Live for Speed racing simulator* and proved to help to mitigate some of driver's mistakes.

Chapter 11

Conclusion

This thesis presents a nonlinear model predictive control strategy which runs in real time, in the order of a few miliseconds. To be precise, the solver safely converges in 15 ms, up to exceptions such as bumping into an obstacle.

The (missing) implementation of the combined force is often a source of inaccuracies in MPCs for cars, as mentioned in [19]. Further work can investigate this aspect, and implement an alternative way of restricting the reaction forces. (The approach on the mind is introducing new optimization variables to represent the reaction forces in the computation of the vehicle states.)

Although the original control strategy — the linear MPC connected with experimentally adjusted PID controllers — can be considered more "primitive," it is better adjusted for control of the accelerator and bake pedal positions. This can be a result of the omitted combined force restriction, or of an inaccuracy in the model for predicting the wheel angular velocities.

Bibliography

[1] D. Efremov, M. Klaučo, T. Haniš, and M. Hromčík, "Driving envelope definition and envelope protection using model predictive control," in *2020 American Control Conference (ACC)*, pp. 4875–4880, 2020.

[2] D. Schramm, M. Hiller, and R. Bardini, *Vehicle Dynamics: Modeling and Simulation.* Springer, Berlin, Heidelberg, 2014.

[3] H. Pacejka, *Tire and vehicle dynamics.* Elsevier, 2005.

[4] D. Efremov, T. Haniš, and M. Hromčík, "Introduction of driving envelope and full-time-full-authority control for vehicle stabilization systems," in *2019 22nd International Conference on Process Control (PC19)*, pp. 173–178, 2019.

[5] Hexagon AB / MSC Software, *Using Adams/Tire. Using the PAC2002Tire Model*, 11 2014. Online.

[6] M. Short, M. J. Pont, and Q. Huang, "Simulation of vehicle longitudinal dynamics," safety and reliability of distributed embedded systems, technical report esl 04–01, Embedded systems Laboratory, University of Leicester, England, 10 2004.

[7] M. Băţăuş, A. Maciac, M. Oprean, and N. Vasiliu, "Automotive clutch models for real time simulation," *Proceedings of the Romanian Academy, Series A: Mathematics, Physics, Technical Sciences, Information Science*, vol. 12, no. 2, pp. 109–116, 2011.

[8] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017.

[9] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[10] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[11] A. Zanelli, A. Domahidi, J. Jerez, and M. Morari, "Forces nlp: an efficient implementation of interior-point methods for multistage nonlinear nonconvex programs," *International Journal of Control*, vol. 93, no. 1, pp. 13–29, 2020.

[12] P. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent, "A fully asynchronous multifrontal solver using distributed dynamic scheduling," *SIAM Journal on Matrix Analysis and Applications*, vol. 23, no. 1, pp. 15–41, 2001.

[13] "Hsl. a collection of fortran codes for large scale scientific computation." http://www.hsl.rl.ac.uk/.

[14] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Rev.*, vol. 47, pp. 99–131, 2005.

[15] H. Ferreau, S. Almér, R. Verschueren, M. Diehl, D. Frick, A. Domahidi, J. Jerez, G. Stathopoulos, and C. Jones, "Embedded optimization methods for industrial automatic control," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 13194–13209, 2017. 20th IFAC World Congress.

[16] Embotech AG, Giessereistrasse 18, CH-8005 Zürich, *FORCESPRO User Manual*, 10 2020. Version 4.1.0.

[17] GAMS Software GmbH, Augustinusstr11b, 50226 Frechen Deutschland, *GAMS User Manual*. Online, retrieved on 2021-01-14. https://www.gams.com/latest/docs/UG_NLP_GoodFormulations.html.

[18] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[19] R. Quirynen, K. Berntorp, and S. Di Cairano, "Embedded optimization algorithms for steering in autonomous vehicles based on nonlinear model predictive control," in *2018 Annual American Control Conference (ACC)*, pp. 3251–3256, 2018.

[20] C. E. Beal and J. C. Gerdes, "Model predictive control for vehicle stabilization at the limits of handling," *IEEE Transactions on Control Systems Technology*, vol. 21, no. 4, pp. 1258–1269, 2013.

[21] T. Twardzik, "Verification platform development for vehicle control system validation," B.S. thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, 2020.