A Nonlinear Model Predictive Control based Evasive Manoeuvre Assist Function

Gijs van Lookeren Campagne

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

A Nonlinear Model Predictive Control based Evasive Manoeuvre Assist Function

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Gijs van Lookeren Campagne

June 21, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

The work in this thesis was supported by Volvo Car Corporation. Their cooperation is gratefully acknowledged.

Abstract

In recent years, research in the field of autonomous driving has been subject to a significant increase in interest. Along with the advances in automation and electrification, cars are slowly transitioning to computers on wheels. The ever-increasing processing power in vehicles gives rise to the implementation of advanced control strategies. The increase in computational power not only allows for novel solutions to automated driving but can also be used for advanced control systems aimed at collision avoidance. At the same time, significant advances have been made towards efficient nonlinear optimisation methods, giving rise to new possibilities for real-time applications.

This thesis focuses on an Evasive Manoeuvre Assistance Function (EMA) to avoid obstacles at the limits of handling using a Model Predictive Control (MPC) approach. Vehicles at the limits of handling typically operate in the nonlinear region of the tyre. For this reason, a Nonlinear Model Predictive Control (NMPC) approach is employed. A baseline scenario with a lateral displacement of 2 metres is considered, representing a common near rear-end collision situation. An optimal trajectory and steering sequence is computed by the MPC over a defined horizon. As MPC considers a model to calculate an optimal control input, a nonlinear two-track model with a simplified Magic Formula is used.

The main drawback of real-time NMPC has always been its computational burden. For the baseline scenario, the limits of real-time NMPC are explored, using the state-of-the-art nonlinear optimisation methods: Sequential Quadratic Programming (SQP) and Interior-Point Method (IPM). Both methods are compared in simulation and an SQP approach is tested on an embedded system within a test vehicle, subject to the disturbances and uncertainties that come with physical driving.

It was found that the proposed NMPC strategies make for a robust yet aggressive manoeuvre with high lateral accelerations, utilising the nonlinear operating region of the tyre. Furthermore, computation times on an embedded system in a test vehicle were found to be sufficiently low for real-time application.

Table of Contents

List of Figures

List of Tables

Preface

MPC is a control strategy that found its origin in the 1970s in the process industry. Known as a computationally expensive control method, it was mainly used for slow processes. In recent years, due to advances in computing power and efficient novel solvers, MPC has become a more popular solution for applications with shorter sampling times. One of the novel fields subject to ongoing experiments with MPC, is the automotive industry.

In March 2019, a Tesla Model 3 crashed into a truck, with autopilot enabled. The roof separated from the car and the driver was killed during the accident. The circumstances of this crash are similar to another fatal autopilot crash, in 2016. In both scenarios, a truck crossed the Tesla's path. Travelling at 110 km/h, the vehicle failed to execute an evasive manoeuvre, as stated by the National Transportation Safety Board (NTSB).

Volvo's CEO, Håkan Samuelsson, stated that in order to launch autonomous driving successfully, an enormous amount of computational power is needed. Volvo aims at having a level 4 autonomous vehicle on the road by 2021, according to the SAE (Society of Automotive Engineers) autonomous driving guideline. At level 4, no driver intervention is needed to ensure safety. Volvo has also put out its so-called Vision 2020 statement in which it states that no one is to be seriously injured or killed by 2020. To achieve this, various Advanced Driver Assistance Systems (ADAS) systems are put in place so that the vehicle is able to avoid accidents by intervening in emergency scenarios.

In October 2018 the partnership between computer hardware manufacturer NVIDIA and Volvo came about with plans to work together on solutions to autonomous driving. To take autonomous driving to the next level, powerful integrated computers need to developed. This powerful processing power opens doors not only to novel image processing solutions for autonomous driving, but also to more advanced control strategies to control the car. These advanced control strategies can then be used to create or improve existing ADAS.

This thesis endeavours to use the newly available processing power for efficient advanced control strategies for an emergency manoeuvre, at the limits of handling. The thesis work was carried out in the Vehicle Motion & Control department at Volvo Cars, Gothenburg. I would like to thank my supervisor at Volvo, Derong Yang and my supervisor in Delft, Manuel Mazo Espinosa for their assistance during this thesis.

Chapter 1

Introduction

1-1 Motivation

In 94% of all vehicle crashes the critical pre-event that led to the crash was caused by the driver of the vehicle [5]. The critical reason is often due to a recognition, decision or performance error of the driver. In [6] a driving simulator test was carried out for a near collision situation. It was found that in 50% of the accidents a swerving manoeuvre should have been initiated. In most cases however, the driver resorted to braking only. Today's driver assistance systems are able to assist drivers through warning signals or braking assistance. For scenarios where solely braking is not sufficient, there are currently no assistance systems in place that allow for complete collision avoidance.

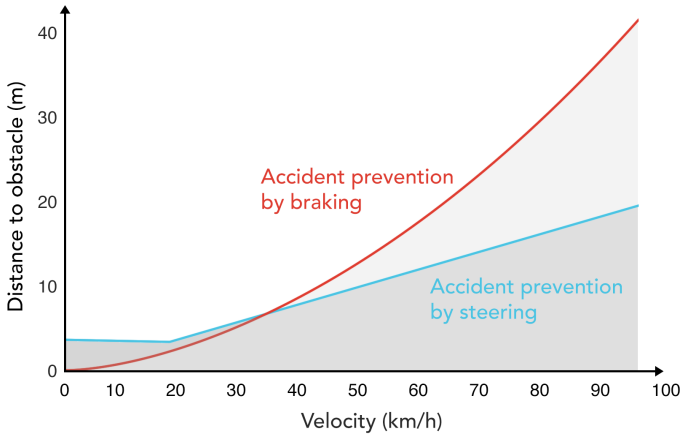


Figure 1-1: Braking and steering distance compared [1]

In figure 1-1 the effectiveness of braking and steering is compared for collision avoidance. This figure shows the minimum needed braking and steering distance. Here, a sigmoid trajectory is considered with a 2 m lateral displacement. It can be seen that at low speeds, solely braking

is sufficient and is in fact a better way to prevent an accident than steering. At higher speeds, the minimum needed distance to avoid the obstacle is lower than for braking. This goes to show that there is a lot to gain when collision avoidance problems are not only considered from a longitudinal (braking) perspective, but also from a lateral (steering) perspective.

An evasive manoeuvre is a manoeuvre that is performed in near collision situations. It involves a sudden steering movement in order to displace laterally and avoid collision with an object. Evasive manoeuvres are typically initiated to avoid rear-end or side collisions with oncoming obstacles. A rear-end collision is a situation where a following vehicle crashes into a leading vehicle. Often these types of collision are caused by a sudden deceleration of the leader or a rapid acceleration of the follower. Almost a third of all accidents are caused by rear-end collision [7]. A side collision is a collision where two vehicles collide into each other, approaching each other perpendicularly. These type of situations often occur at intersections and parking lots. An evasive manoeuvre aims at avoiding these type of collisions. A typical trajectory for an evasive manoeuvre in a rear-end collision scenario can be seen in figure 1-2.

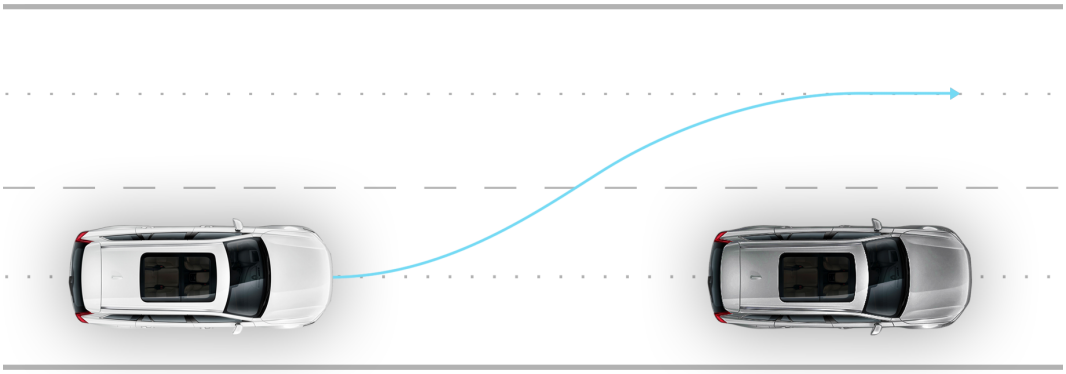


Figure 1-2: Evasive manoeuvre trajectory

The benefit of having driver assistance systems in critical handling situations can be observed in figure 1-3. This figure shows a friction circle for an aggressive left turn. Two types of driver behaviour are shown: amateur and professional. A friction circle shows how much longitudinal and lateral friction a driver can obtain. It can be seen that a professional driver is able to drive much closer to the limits of handling than an amateur driver does. In other words, amateur drivers are not able to obtain all available friction in critical situations. Assisting drivers in performing critical manoeuvres might therefore be very beneficial to avoid dangerous collisions. Currently, Electronic Stability Control (ESC) is an example of a widely implemented driver assistance system to help drivers maintain stability. For single vehicle crashes, it is found that fatality risks are reduced by 30% for passenger cars equipped with ESC [8].

For an evasive manoeuvre, the driver assistance function can be referred to as an Evasive Manoeuvre Assistance Function (EMA). Current EMA functions typically rely on a bang-bang controller with a sequence of predefined assistance steering torque. This sequence consists of constant positive torque assistance, followed by constant negative steering torque. The amplitudes of the provided constant torque is around 3-5 Nm. To initiate the function, a threat needs to be detected. Furthermore, the driver applied steering wheel torque needs to

be above a certain threshold value.

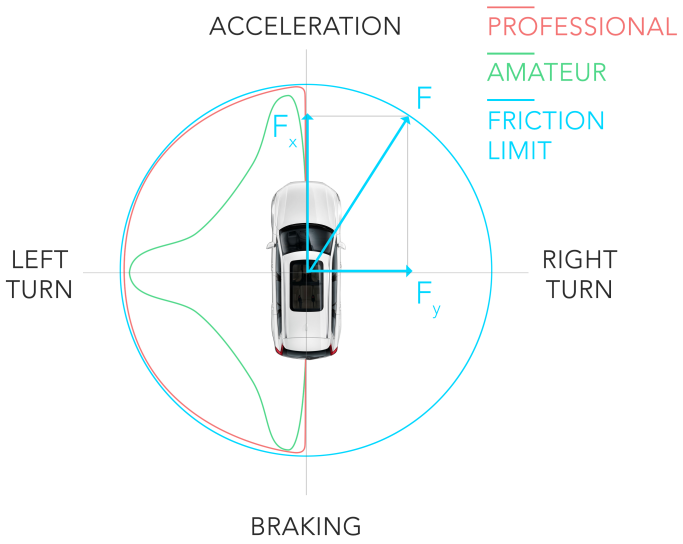


Figure 1-3: Friction circle for different kinds of driving behaviour [2]

In order to provide a driver assistance system, a control system needs to be put in place. In this thesis, the application of Model Predictive Control (MPC) is researched to solve the evasive manoeuvre problem. In recent years MPC control strategies have seen a significant increase in usage for numerous new applications, among which the automotive industry [9]. The recent spike in interest to use MPC for new applications can be ascribed to the ever increasing processing power in modern day computers. This allows to solve complex optimisation problems, that were not deemed to be solvable before.

1-2 Problem definition

Rear-end crashes between two vehicle are the type of crashes that occur most frequently [7]. As can be seen in figure 1-4, the obstacle width denotes the lateral displacement needed to avoid the obstacle. It is assumed that using today's sensor fusion methods, the obstacle's width can be measured. This obstacle width serves to create the reference displacement which is used as an input to the MPC. On average, cars are around 2 m wide [10]. In order to be able to compare different methods and controllers for the same scenario, a baseline obstacle width of 2 m is considered throughout this thesis.

The evasive manoeuvre scenario can therefore be described as reaching a lateral reference position of 2 m as fast as possible, with a minimal longitudinal displacement X_s, as shown in figure 1-5. In practice, this means that the function needs to be triggered by a set of triggering conditions. The function then executes a manoeuvre based on the obstacle's width.

In evasive manoeuvres, cars typically tend to be near the limits of handling. This means that the tyres operate near their friction limits. In these conditions, the relation between the lateral tyre forces and the slip of the tyre, is nonlinear. Furthermore, the vehicle dynamics

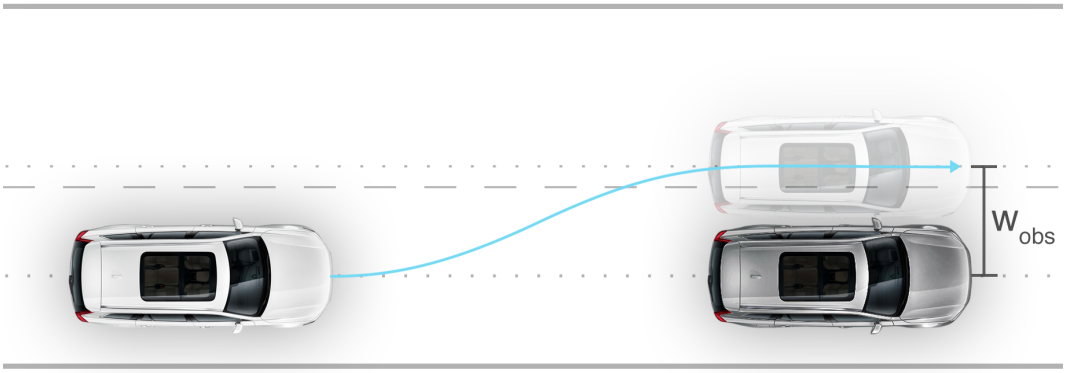


Figure 1-4: Reference displacement

are inherently nonlinear. To cope with these nonlinearities, a Nonlinear Model Predictive Control (NMPC) strategy is considered in this thesis.

In this thesis, the EMA problem will be approached from a fully autonomous perspective. In other words, no driver applied steering wheel torque is taken into consideration.

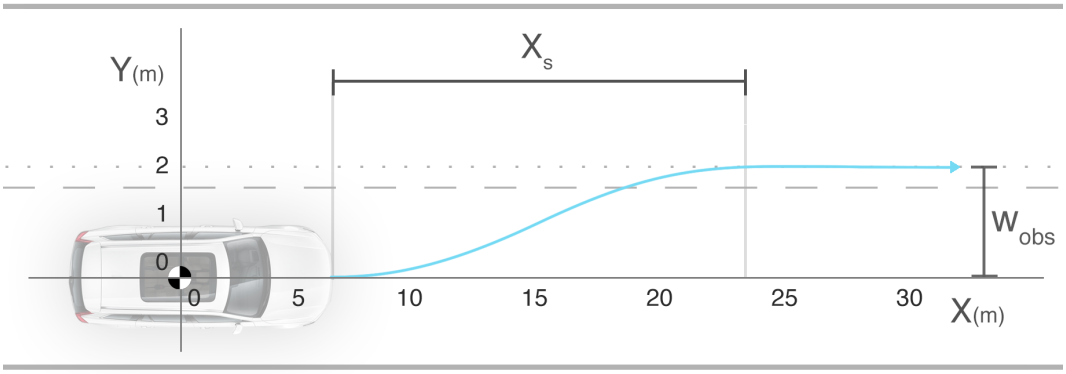


Figure 1-5: Baseline scenario

The objective of this thesis is:

To develop a real-time NMPC strategy to laterally displace 2 m within the shortest possible longitudinal displacement.

Based on the literature study, a set of scenario targets is formulated:

Table 1-1: Scenario targets

Scenario target	Value
Sampling frequency	25 Hz
Lateral acceleration	6-9 m/s^2
Longitudinal distance travelled	20-25 m
Maximum allowable overshoot	0.5 m

The main novelty of this master thesis is the focus on solving the NMPC problem online. By regarding state-of-the art solvers, the limits of real-time NMPC are explored. It is assessed whether this approach is feasible in terms of computational burden. The designed MPC is then tested in the virtual test environment CarMaker before being tested on a rapid prototyping dSPACE system within a physical vehicle.

1-3 Major contributions

This thesis aims to include the following major contributions to the field:

- Analysis of the current possibilities of NMPC for limit handling scenarios, with a focus on dealing with computational complexity.
- Development and testing of a real-time NMPC control strategy on an embedded system within a physical car.
- Comparison of the two major Nonlinear Programme (NLP) solvers for NMPC.

1-4 Thesis outline

In chapter 2, a summary is presented of the literature survey carried out prior to this thesis. Here, emphasis is put on the current state-of-the-art researches in the field and a decision tree is shown to define the main approach of this thesis. MPC relies on a model of the system dynamics to define a control input. The nonlinear vehicle and tyre model used in the MPC are discussed in chapter 3. After having defined the model within the MPC, an introduction to MPC is given in chapter 4. In this chapter, the general formulation for MPC is explained and it advantages over other control strategies are discussed. To solve a nonlinear MPC problem, NLP methods need to be regarded. The two most renowned methods for solving nonlinear MPC problems are explained in chapter 5. In chapter 6 and 7 the results of the CarMaker simulation and experimental test runs are discussed, respectively. Finally, based on the findings in this thesis, a conclusion will be drawn in chapter 8.

Chapter 2

Literature survey

2-1 Current state-of-the-art

Previous research for limit handling [11] [12] and obstacle avoidance scenarios [13] [4] [14] [15] [16] [17] [18] [19] relied on linearisation of the vehicle dynamics in order to cope with the computional burden. In [20] only the tyre model was linearised. In these researches, in one way or another, concessions were made in terms of model accuracy. A Nonlinear Model Predictive Control (NMPC) approach was employed in [21]. For an obstacle avoidance scenario in icy conditions, an obstacle was avoided successfully at $v_x = 10\,\mathrm{m/s}$, with low computation times. In this research, the proposed Model Predictive Control (MPC) scheme was only tested in simulations.

Experimental tests introduce a novelty to this thesis as there have been few experimental tests in past research for NMPC based evasive manoeuvre type scenarios. Most research involves simulation only. The additional uncertainties that come with experimental tests (e.g. inaccurate measurement signals) introduce yet another degree of complexity.

There has been some research using NMPC for similar scenarios that also includes experimental tests: In [22] a NMPC approach was used successfully for a double lane change over 150 m longitudinal displacement which did not involve limit handling. In [23] a NMPC approach was used for a double lane change manoeuvre in icy conditions. Here, the commercial NPSOL package was used, using a Sequential Quadratic Programming (SQP) method. It was found that at 17 m/s, a similar velocity as regarded in this thesis, simulation computation times were far too long (1.3 s) to perform in real-time. For experimental tests at 10 m/s the controller was not able to stabilise and the vehicle would start to skid. In this case, the MPC solver failed to come up with a feasible solution.

Based on the literature study it is concluded that as of today, an evasive manoeuvre scenario using MPC with nonlinear vehicle and tyre dynamics, has never been implemented successfully on an experimental setup before. In previous research, the dynamics were either linearised, or the propsed MPC was not tested on an experimental setup. A thorough search of the relevant literature did not yield any researches comparable to this thesis.

2-2 Design choices

A schematic framework of the design choices made in the literature survey can be observed in figure 2-1. The path denoted in red will be the main approach to solve the problem in this thesis. On the left the four main stages of the literature study are shown. The options that were considered for these stages are listed on the right. The main theme during the literature survey was based on the trade-off between computational complexity and accuracy.

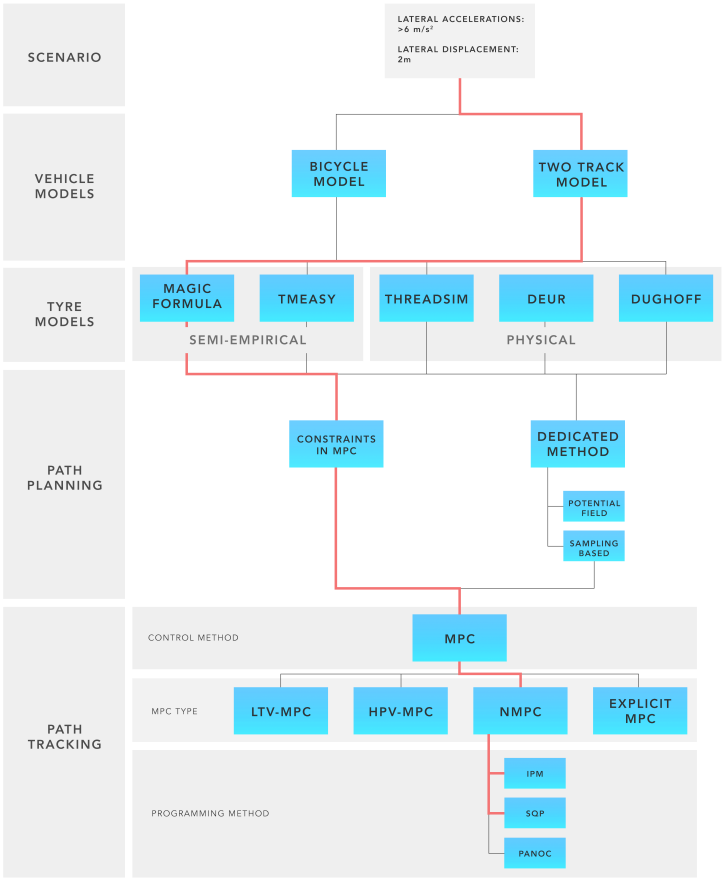


Figure 2-1: Overview of literature study outcome

Based on past research, the vehicle models that were considered are a bicycle model and a simple two-track model. Both models have three degrees of freedom. As these models share a lot of similarities, the complexity of both models is similar. Therefore, the slightly more accurate two track model is used.

In terms of tyre models Pacejka's Magic Formula [24] remains unrivalled in terms of accuracy and computational complexity. Its major downside is the large amount of parameters that need to be determined.

Path planning and path tracking will be carried out by the MPC. Dedicated path planning methods are more suitable for dynamic and complex driving environments. The main scenario

in this thesis does not benefit of any complex dedicated path planning method. A cost function within the MPC allows to define a destination and penalise any movement away from this destination such that the vehicle moves to the destination.

The MPC problem is solved by means of the two most renowned Nonlinear Programme (NLP) methods for NMPC: SQP and Interior-Point Method (IPM). Both these methods are tested and compared extensively in simulation runs in a highly realistic CarMaker simulation environment. Finally, an SQP method is implemented on an embedded dSPACE MicroAutobox II system in a test vehicle.

Chapter 3

System dynamics

As Model Predictive Control (MPC) is a model-based control strategy, a model is needed to predict future outputs of the system. Based on a literature survey, a 3 degree-of-freedom nonlinear two-track model was selected. It was found that this model provides a balanced trade-off between computational complexity and model accuracy. A nonlinear tyre model was used to be give a more accurate description of the tyre dynamics at large slip angles. During limit handling, cars typically tend to be in the nonlinear region, where slip angles are large. The difference between a linear tyre model and a nonlinear tyre model can be observed in figure 3-1.

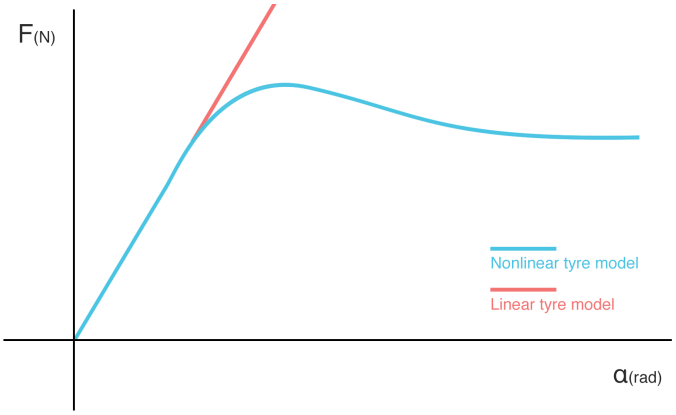


Figure 3-1: Comparison between nonlinear and linear tyre model

The tyre model that is used is Pacejka's Magic Formula [24]. This empirical formula describes the relation between the slip angle and slip ratio to the lateral and longitudinal tyre force, respectively. Based on the literature survey it was found that this tyre model provides the best balance in terms of computational complexity and accuracy.

3-1 Vehicle coordinate frames

In figure 3-2 the different coordinate frames are shown. The global, local and wheel coordinate frames are denoted by C_0, C_1 (red) and C_2 (blue) respectively. Green denotes the velocities at the front wheel. Due to the fact that only the front wheel turn when steering, the velocity of the front wheels needs to be mapped to the wheel coordinate frame.

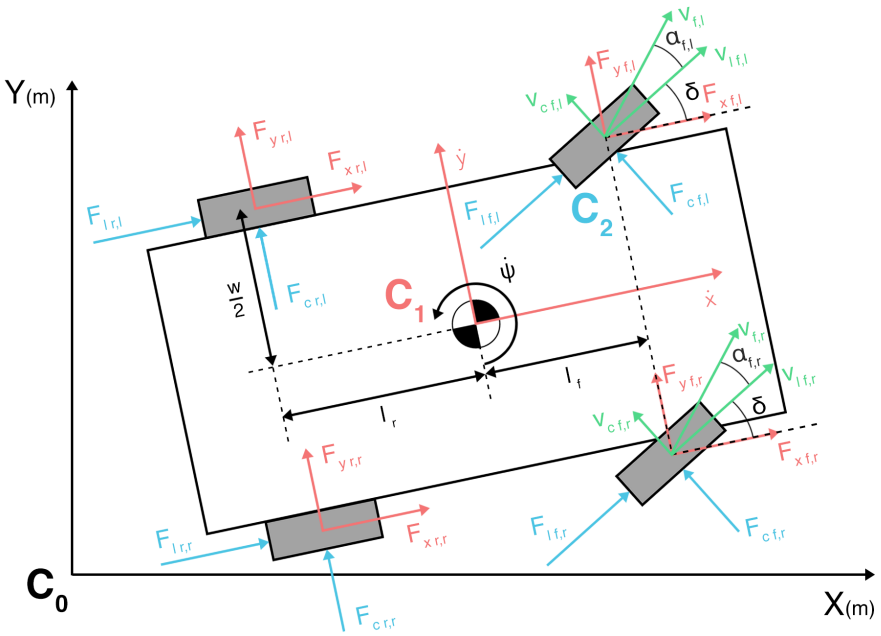


Figure 3-2: Vehicle coordinate frames

3-2 Vehicle model

In this thesis, two scenarios are considered. The first scenario is a steering only scenario, where only a steering input is considered. In the second scenario, the brakes at the individual wheels are added as inputs such that a combined steering and braking manoeuvre is carried out. Both these scenarios incorporate different vehicle models. For both scenarios the following equations of motion in the vehicle coordinate system (C_1) serve as the basis of the vehicle model [25]:

$$\begin{aligned} m\ddot{x} &= m\dot{y}\dot{\psi} + F_{x_f,l} + F_{x_f,r} + F_{x_r,l} + F_{x_r,r} \\ m\ddot{y} &= -m\dot{x}\dot{\psi} + F_{y_f,l} + F_{y_f,r} + F_{y_r,l} + F_{y_r,r} \\ I_z\ddot{\psi} &= l_f(F_{y_f,l} + F_{y_f,r}) - l_r(F_{y_r,l} + F_{y_r,r}) + \frac{w}{2}(-F_{x_f,l} + F_{x_f,r} - F_{x_r,l} + F_{x_r,r}) \end{aligned} \tag{3-1}$$

This vehicle model incorporates longitudinal, lateral and yaw degrees-of-freedom. They are expressed as $\ddot{x}$, $\ddot{y}$ and $\ddot{\psi}$, respectively. The parameters l_f and l_r denote the respective longitudinal distances from the centre of gravity to the front and rear axle. The track width is

denoted by w. The yaw moment of inertia is denoted by I_z and the mass of the vehicle is displayed as m. The longitudinal and lateral tyre forces are expressed as F_{x_i} and F_{y_i}, in the C_1 frame.

The velocities in the global coordinate coordinate frame (C_0) can be mapped from the vehicle coordinates (C_1):

$$\begin{aligned} \dot{Y} &= \dot{x}\sin\psi + \dot{y}\cos\psi \\ \dot{X} &= \dot{x}\cos\psi - \dot{y}\sin\psi \end{aligned} \tag{3-2}$$

The lateral (F_{y_i}) and longitudinal forces (F_{x_i}) in the vehicle coordinate frame are mapped from the tyre forces in the wheel coordinate frame (C_2) to the vehicle coordinate frame (C_1).

$$\begin{aligned} F_{y_i} &= F_{l_i}\sin\delta + F_{c_i}\cos\delta \\ F_{x_i} &= F_{l_i}\cos\delta - F_{c_i}\sin\delta \end{aligned} \tag{3-3}$$

where F_{l_i} and F_{c_i} denote the longitudinal and lateral tyre forces in the C_2 coordinate frame, respectively. The steering angle δ is only considered for the front wheels. For the rear wheels, δ is always equal to 0.

3-2-1 Vehicle model for steering only scenario

For the steering only scenario the system dynamics of (3) are used with 7 states and 1 input, as shown in (3-4). The steering rate is implemented as an input, whereas the steering angle enters the system as a state. This allows to incorporate the actuator limits, for which both the steering rate and steering angle need to be constrained. MPC allows to put constraints on the inputs and states. If the steering angle is set as the input, only the angle can be constrained directly, but not its derivative. The state vector x and control input u are defined as:

$$x = \begin{bmatrix} X \\ Y \\ \dot{x} \\ \dot{y} \\ \psi \\ \dot{\psi} \\ \delta \end{bmatrix}, \quad u = \dot{\delta} \tag{3-4}$$

Here, the global positions X and Y (as opposed to local) are used as they will be of interest when calculating an optimal control input such that a particular position is reached. The state vector x is not to be confused with a local vehicle position x (C_1), which is not considered in this thesis.

For the steering only scenario, only lateral dynamics are implemented as longitudinal dynamics are relatively insignificant if no braking is applied. To simplify the model and therefore reduce computational complexity, a constant deceleration $\ddot{x} = c$ is used to account for the decrease in velocity. Thus, the longitudinal dynamics shown in 3-1 are omitted and replaced by a constant deceleration $\ddot{x} = c$.

3-2-2 Vehicle model for steering and braking scenario

In this scenario, the brake torques are added as inputs at each wheel. The brake torques T_{b_i} enter the wheel equations of motion (3-5), for which the dynamics are given by [25]:

$$\begin{aligned} I_{d_{f,l}}\dot{\omega}_{f,l} &= -F_{l_{f,l}}r - T_{b_{f,l}} - B_d\omega_{f,l} \\ I_{d_{f,r}}\dot{\omega}_{f,r} &= -F_{l_{f,r}}r - T_{b_{f,r}} - B_d\omega_{f,r} \\ I_{d_{r,l}}\dot{\omega}_{r,l} &= -F_{l_{r,l}}r - T_{b_{r,l}} - B_d\omega_{r,l} \\ I_{d_{r,r}}\dot{\omega}_{r,r} &= -F_{l_{r,r}}r - T_{b_{r,r}} - B_d\omega_{r,r} \end{aligned} \tag{3-5}$$

Here, I_{d_i} denotes the moment of inertia of each wheel. The angular velocities of the wheels are given by ω_i. The wheel radius is displayed as r. The driveline damping coefficient is expressed as B_d. These equations of motion, combined with the dynamics shown in (3-1) represent the dynamics used for the steering an braking scenario. Here it is assumed that no torque is provided by the engine. Naturally, braking has a significant impact on the longitudinal dynamics. Therefore, the longitudinal dynamics of (3-1) are no longer omitted.

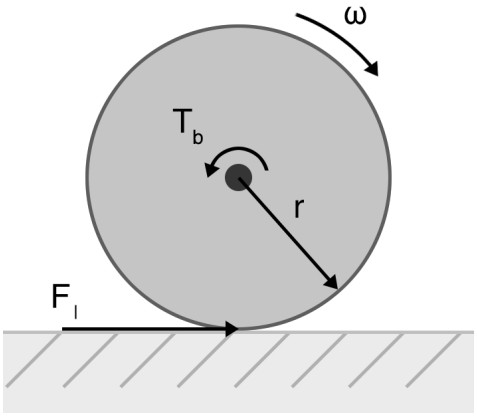


Figure 3-3: Longitudinal wheel dynamics

The total amount of states and inputs for this scenario is then given by:

$$x = \begin{bmatrix} X \\ Y \\ \dot{x} \\ \dot{y} \\ \psi \\ \dot{\psi} \\ \delta \\ \omega_{f,l} \\ \omega_{f,r} \\ \omega_{r,l} \\ \omega_{r,r} \end{bmatrix}, \quad u = \begin{bmatrix} \dot{\delta} \\ T_{b_{f,l}} \\ T_{b_{f,r}} \\ T_{b_{r,l}} \\ T_{b_{r,r}} \end{bmatrix} \tag{3-6}$$

3-3 Nonlinear tyre model

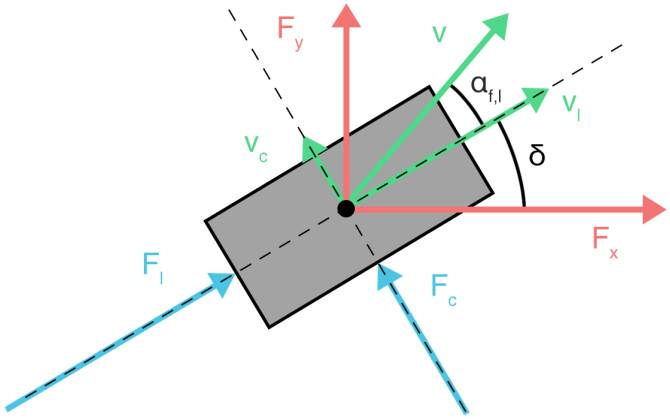


Figure 3-4: Slip angle

The slip angle at each wheel is defined as the angle between the direction in which the wheel is travelling and the angle in which it is pointing:

$$\alpha_i = \tan^{-1} \frac{v_{c_i}}{v_{l_i}} \tag{3-7}$$

The longitudinal and lateral velocities in the wheel coordinate frame are given by:

$$\begin{aligned} v_{l_i} &= v_{y_i} \sin\delta + v_{x_i} \cos\delta \\ v_{c_i} &= v_{y_i} \cos\delta - v_{x_i} \sin\delta \end{aligned} \tag{3-8}$$

where the velocities in the vehicle frame are:

$$\begin{aligned} v_{y_{f,l}} &= \dot{y} + a\dot{\psi} \quad & v_{x_{f,l}} &= \dot{x} - c\dot{\psi} \\ v_{y_{f,r}} &= \dot{y} + a\dot{\psi} \quad & v_{x_{f,r}} &= \dot{x} + c\dot{\psi} \\ v_{y_{r,l}} &= \dot{y} - b\dot{\psi} \quad & v_{x_{r,l}} &= \dot{x} - c\dot{\psi} \\ v_{y_{r,r}} &= \dot{y} - b\dot{\psi} \quad & v_{x_{r,r}} &= \dot{x} + c\dot{\psi} \end{aligned} \tag{3-9}$$

The lateral tyre force is given by Pacejka's Magic Formula [24]:

$$F_{c_i} = -D \sin(C \tan^{-1}((1-E)B\alpha_i + E \tan^{-1}(B\alpha_i))) \tag{3-10}$$

where the coefficient D denotes the peak tyre force:

$$D = AF_{z_i} \tag{3-11}$$

The lateral tyre forces depend on the normal loads at each wheel. During a manoeuvre, the normal loads change due to load transfer. Load transfer dynamics depend on the lateral acceleration a_y, which is not included in the states. To prevent an algebraic loop, extra states need to be added to model the load transfer dynamics [21]. Therefore, to keep the computational complexity down, a static load is assumed:

$$\begin{aligned} F_{z_f} &= \frac{mgl_r}{2(l_f + l_r)} \\ F_{z_r} &= \frac{mgl_f}{2(l_f + l_r)} \end{aligned} \tag{3-12}$$

The slope at $\alpha = 0$ is given by the slope of the product BCD. This slope is equal to the cornering stiffness used in linear tyre models. The parameter E influences the curvature near the peak. The tyre is saturated at a slip angle a_{sat}. At this slip angle, the peak tyre force D is obtained.

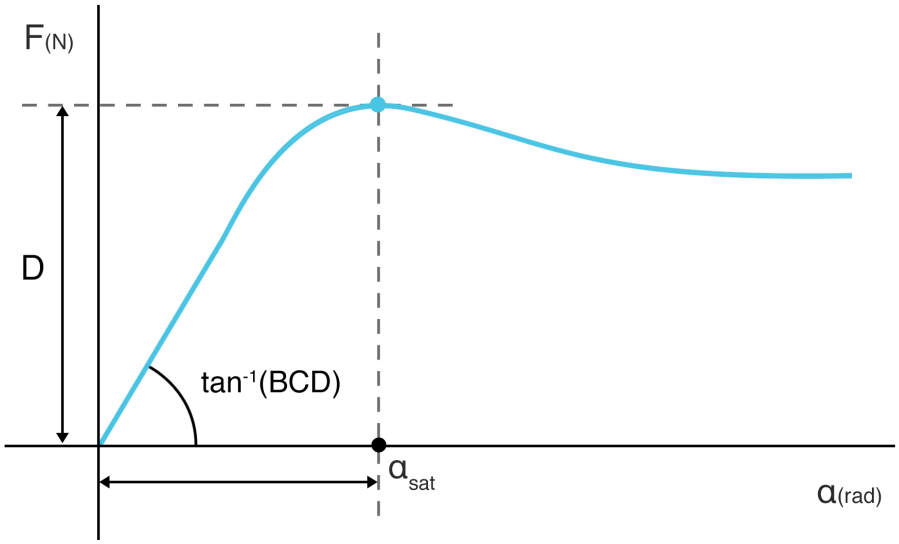


Figure 3-5: Pacejka's Magic formula

Similar as for the lateral tyre force, a relation exists between the slip ratio κ and longitudinal tyre force:

$$F_{l_i} = -D \sin(C \tan^{-1}((1 - E)B\kappa_i + E \tan^{-1}(B\kappa_i))) \tag{3-13}$$

where the slip ratio κ (in the case of braking) is given by:

$$\kappa = \frac{r\omega_i}{v_{l_i}} - 1 \tag{3-14}$$

Chapter 4

Model Predictive Control

4-1 What is Model Predictive Control (MPC)?

In Model Predictive Control (MPC) a cost function is optimised such that a reference is tracked over a horizon N. Current time step measurements and future time step predicted outputs are used to track the reference over the horizon. The predicted outputs are predicted using a model of the system. By minimising a cost function, a sequence of optimal control inputs is calculated over a horizon. The current time step control input is then implemented at each time step. This process is then reiterated for the next time step. The horizon shifts along with the current time step and can therefore be referred to as a receding horizon. The predictive nature of MPC allows to account for future time steps.

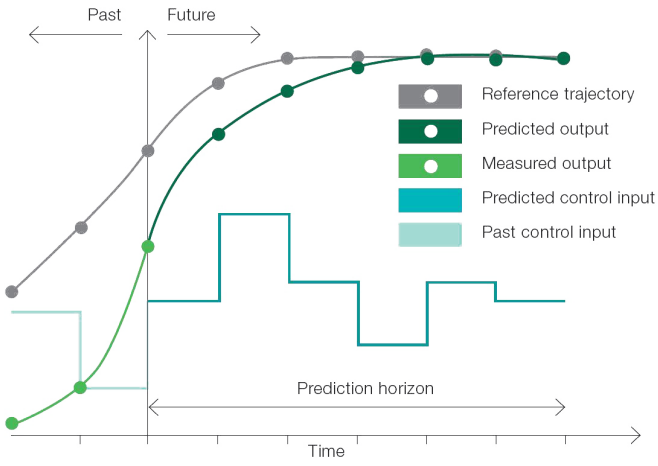


Figure 4-1: MPC reference tracking [3]

To define an objective a cost function is used to describe the desired behaviour. In the cost

function, the difference between the reference states and the measured and predicted states is minimised over a horizon. A general formulation for an MPC optimisation problem is:

$$\begin{aligned} \min_{x(k),u(k)} \quad & J(x(k),u(k)) \\ \text{subject to} \quad & x(k) \in \mathcal{X}, u(k) \in \mathcal{U} \\ & x(k+1) = f(x(k),u(k)) \end{aligned} \tag{4-1}$$

where the states and inputs can be contained within sets $\mathcal{X}$ and $\mathcal{U}$:

$$\begin{aligned} \mathcal{X} &:= \{x \in \mathbb{R}^n \mid x_{\min} \leq x \leq x_{\max}\} \\ \mathcal{U} &:= \{u \in \mathbb{R}^m \mid u_{\min} \leq u \leq u_{\max}\} \end{aligned} \tag{4-2}$$

Furthermore, the optimisation problem is subject to the system dynamics $f(x(k),u(k))$. The general cost function for MPC is as follows:

$$J = ||x(N) - x_r(N)||_P^2 + \sum_{k=1}^{N} ||x(t+k) - x_r(t+k)||_Q^2 + ||u(t+k) - u_r(t+k)||_R^2 \tag{4-3}$$

where x_r and u_r denote the reference state vector and control input. The current time step is t and the horizon is denoted by N. The iterations for which the output is predicted and the cost function is minimised are $k = 1, \ldots, N$. The weight matrices Q and R specify the weights on tracking the reference states and penalising the input, respectively:

$$Q = \begin{bmatrix} q_1 & 0 & \ldots & 0 \\ 0 & q_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & q_n \end{bmatrix}, \quad R = \begin{bmatrix} r_1 & 0 & \ldots & 0 \\ 0 & r_2 & \ldots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & 0 & 0 & r_m \end{bmatrix} \tag{4-4}$$

The individual weights on each state and input are denoted by $q_1 \ldots q_n$ and $r_1 \ldots r_m$, respectively. A higher weight indicates that tracking a certain state or penalisation of an input is deemed more important.

Essentially, an MPC control strategy is very much analogous to driving a car [9]. The driver has a reference trajectory in mind over a certain horizon (e.g. staying within a road lane). Based on previous experiences of driving, the driver has a mental model of the car and know how the car will behave when applying a certain input. Naturally, the control input is only applied at the current time step but is also planned ahead (e.g. planning to brake before an upcoming turn). This is in contrast to classical control strategies such as PID, where only past and current errors are taken into account [9].

4-2 Motivation for MPC

Over the recent years, MPC has become a more attractive solution for automotive applications [9]. This can be ascribed to numerous reasons. First off, MPC allows to implement constraints. It is therefore easy to take into account e.g. actuator limits in the optimisation problem. Secondly, tuning effort is low as a model in considered in the optimisation itself. This model is considered in the cost function and allows to simply put weights on objectives that are deemed important. Lastly, MPC naturally is a robust type of control as an optimal solution is calculated for each time step. When a disturbance interferes, this will be accounted for in the optimal control input that is calculated in the next time step.

The major drawback of MPC is its computational complexity. Complex (nonlinear) system dynamics and an optimisation problem over an entire horizon need to be taken into account. Compared to more conventional control methods like Proportional-Integral-Derivative control (PID) control, MPC comes with a higher computational burden. Due to the trend towards automation within vehicles, processors within vehicles become more powerful each iteration. Volvo Cars for instance, have joined forces with NVIDIA to supply advanced computers geared towards AI solutions for autonomous driving [26]. This advance in available computational power within cars will also allow to cope with MPC's computational burden better over the next few years.

In [4] a comparison is drawn between multiple control methods for a collision avoidance scenario. An overview of its findings is shown in table 4-1. It was found that PID control requires a lot of effort to tune right for complex systems and the actuator limits are not taken into account. Robust control is a control strategy where uncertainties are taken into account. Robust control shows a trade-off between performance and robustness, and its performance is insufficient for collision avoidance applications. Input-output linearisation has the advantage that the performance and stability of the controller can be analysed easily. For evasive manoeuvres at the limits of the actuators with nonlinear dynamics, this type of control is not suitable [4].

MPC allows to include constraints (such as actuator constraints, state constraints, maximum feasible tire forces) which gives it an edge over other methods [27]. It is able to handle complex models well and therefore less tuning effort is needed, as a lot of the model information is considered. Tuning is mostly a matter of adjusting weights in the cost function. Tuning the cost function provides a straight-forward and intuitive way of achieving the desired performance. Furthermore, it is able to cope with model errors as planned trajectories are replanned at every time step [4]. MPC's major drawback compared to other methods, is its computational complexity.

	Robust Control	**PID control**	**I/O Linearisation**	**MPC**
Vehicle Dynamics	∘	−	∘	+
Actuator Limits	∘	−	−	+
Computational Complexity	∘	+	∘	−
Customization of Performance	−	+	∘	+
Tuning Effort	+	−	∘	+

Table 4-1: Overview of control methods for collision avoidance manoeuvres [4]

4-3 Solving the MPC problem

The two most established approaches to solve a Nonlinear Model Predictive Control (NMPC) problem are Sequential Quadratic Programming (SQP) and the Interior-Point Method (IPM). In this thesis, both these methods are compared for solving an evasive manoeuvre problem. In section 5 the workings of both methods are be explained. In section 6 both methods are compared in a highly realistic CarMaker simulation environment. Due to the fact that no open source code generation tool for the used Nonlinear Programme (NLP) solver for IPM is available, real-life tests are only carried out for SQP. These can be observed in chapter 7. Based on the comparisons in simulations a conclusion can be made as to whether IPM is a promising method for solving this problem.

4-4 Constraints

Hard constraints can be imposed directly on the states and control inputs. This allows to take into account the actuator limits in the optimisation. As explained earlier in this thesis, this allows to constrain the steering rate, as the actuators in the steering wheel are limited to a rate of 12rad/s. By taking into account the steering ratio, the steering rate of the front wheels can be determined. It is also possible to put constraints on functions of the states and control inputs. In this research the steering rate and steering angle are constrained:

$$\begin{aligned} \delta_{\min} \leq \delta \leq \delta_{\max} \\ \dot{\delta}_{\min} \leq \dot{\delta} \leq \dot{\delta}_{\max} \end{aligned} \tag{4-5}$$

For the combined steering and braking scenario, four additional brake torques are added as control inputs. It was found that the brake torque computed by the MPC is nowhere near the limit of brake torque that can be applied. Braking near the limits while steering aggressively

inevitably makes for an unstable manoeuvre, so high brake torque will not be considered by the MPC. Besides, braking slows down the vehicle, making for a slower manoeuvre and therefore a higher cost as the reference displacement is reached at a later stage. For these reasons, the brake torque remains unconstrained in the optimisation.

Aside from steering, it is also possible to constrain other states, such as the global position Y. However, hard constraining positions is prone to cause computational or infeasibility issues within the optimisation. Therefore it is desired to only implement hard constraints when absolutely necessary. It is preferred to tune the cost function to reach the desired performance. Penalising undesired behaviour in the cost function is known to be significantly more efficient than through hard constraints.

4-5 Implementation

In figure 4-2 the structure of the implemented real-time MPC can be seen. A cost function needs to be defined to minimise the difference between the measured and predicted states (x_m and x) and the reference states (x_r). The measurement states x_m are updated at each time step and serve as an input to the MPC. The optimal control input u^* is the output of the MPC, which is then used as an input to the actuators of the vehicle. Furthermore, the initial conditions for the states x_0 and input u_0 need to be defined within the MPC.

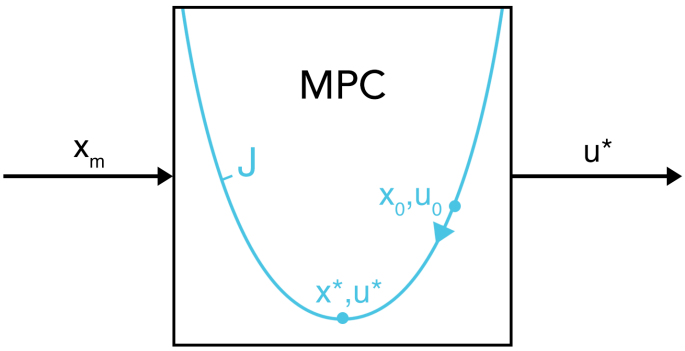


Figure 4-2: MPC structure

Every sampling time step T_s, the states and controls are predicted over a specified time horizon H. This horizon is divided in N control intervals, where one control interval's time step is defined by h. Therefore, the amount of control intervals is defined by $N = \frac{H}{h}$. The computation time should be lower than the sampling time to be able to run in real-time. Therefore, the length of the horizon and the amount of control intervals should be tuned such that this is possible.

As mentioned earlier, the designed MPC is tested in a CarMaker simulation environment as well as in a test vehicle.

For the simulation runs, the measurement states can be obtained directly from CarMaker's own highly realistic vehicle model. Here, no measurement errors are considered that are present in a real vehicle. In CarMaker, a desired steering rate $\dot{\delta}_{des}$ can be requested directly at the front wheels. In other words, no steering wheel model needs to be considered.

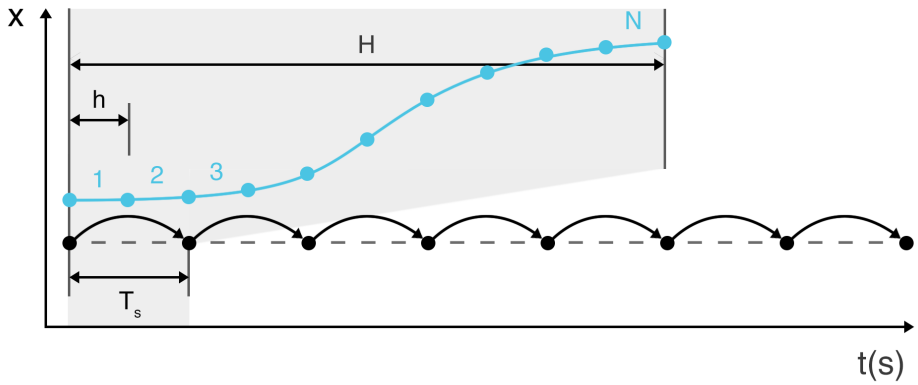


Figure 4-3: Sampling times within MPC

During experimental tests, challenges arise that were not present during simulation runs. First and foremost, C code needs to be built so that the MPC can be employed on the dSPACE system in the test vehicle. Secondly, the measurement signals measured by the Inertial Measurement Unit (IMU) are not as accurate as the measurement signals in the simulation environment. This provides a challenge for the MPC in terms of robustness. Lastly, a sufficiently accurate representation of a steering wheel torque model needs to be derived. This allows to map the requested steering rate to a requested steering torque. This way, the actuator is able to track the desired steering rate computed by the MPC. A schematic overview of the simulation and experimental setup can be observed in 4-4.

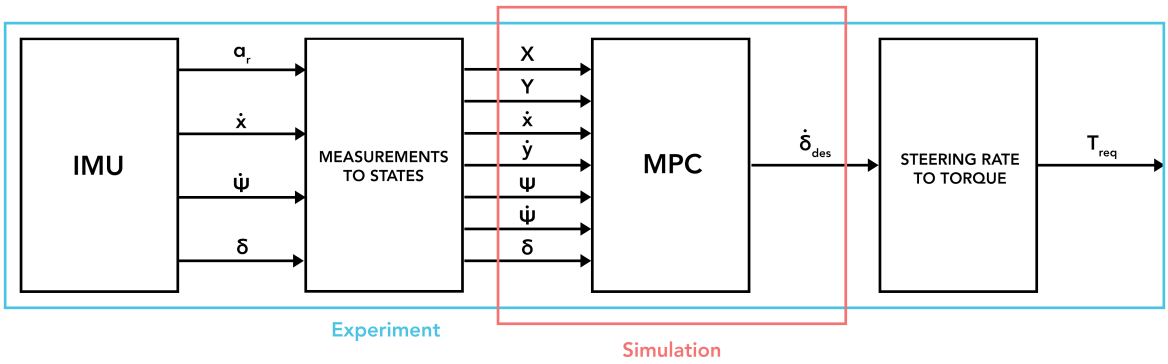


Figure 4-4: Schematic overview of simulation and experimental setup

Chapter 5

Nonlinear programming

To solve a Model Predictive Control (MPC) problem, the optimisation problem (5-1) needs to be cast as a Nonlinear Programme (NLP) (5-2).

$$\begin{aligned} \min_{x(t),u(t)} \quad & J(x(t),u(t)) \\ \text{subject to} \quad & u(t) \in \mathcal{U}, x(t) \in \mathcal{X} \\ & \dot{x} = f(x(t),u(t)) \\ & g_i(x(t),u(t)) \leq 0 \quad \text{for each } i \in \{1,...,m\} \\ & h_j(x(t),u(t)) = 0 \quad \text{for each } j \in \{1,...,p\} \end{aligned} \tag{5-1}$$

The optimisation problem seen in (5-1) needs to be posed as a specific formulation that a dedicated NLP solver is able to solve. A transcription method transforms the optimisation problem to an NLP. In the optimisation problem of (5-1) the decision variables are the functions $x(t)$ and $u(t)$. In an NLP, these functions are transformed to a set of real numbers. The differential equations in (5-1) are transformed to algebraic equations:

$$\begin{aligned} \min_{z(k)} \quad & f(z) \\ \text{subject to} \quad & z \in \mathcal{Z} \\ & g_i(z) \leq 0 \quad \text{for each } i \in \{1,...,m\} \\ & h_j(z) = 0 \quad \text{for each } j \in \{1,...,p\} \end{aligned} \tag{5-2}$$

The optimisation problem is rewritten as a function of a set of decision variables z. Here, z defines the discrete set of state and input points for which the optimisation problem is solved and for which the constraints need to be satisfied. The objective $f(z)$ defines the mapping from decision vector z to cost function J. Once the NLP formulation is obtained, the NLP can be solved by a dedicated NLP solver.

In this chapter, two approaches to solving the MPC problem will be presented. The first approach is an Interior-Point Method (IPM) using the IPOPT [28] solver. Here, the state and control trajectories are transcribed using a direct collocation method. The open-source package CasADi [29] is used to implement and solve the NLP. As explained previously, this approach will only be tested in simulation.

A Sequential Quadratic Programming (SQP) method is implemented using the qpOASES [30] solver in the open-source package ACADO [31]. This approach will be tested in simulation as well as during real-life tests on an embedded dSPACE MicroAutobox II setup in a Volvo V40 test car.

An overview of both approaches can be seen in table 5-1:

Table 5-1: Overview of approaches for solving the NLP

	Solver	**Transcription method**	**Framework**	**Simulation**	**Tests**
I.	IPOPT (IPM)	Direct Collocation	CasADi	Yes	No
II.	qpOASES (SQP)	Multiple Shooting	ACADO	Yes	Yes

CasADi (IPOPT/Direct collocation) is known to be faster for larger problems (a high number of states) [32]. Smaller to medium sized problems are likely to be solved faster by ACADO using SQP and a multiple shooting method. Problem size varies for different experiments throughout this thesis. In practice, this means that lower fidelity vehicle models are likely to perform better using ACADO (and SQP) and higher fidelity models will probably perform better using CasADi (and IPM). SQP is known to be better at handling highly (hard) constrained problems. A comparison between both CasADi and ACADO will provide insight in which framework will suit the scope of this thesis better. Both methods will make use of so-called warm-starts to initialise at each iteration. This means that the previous optimal solution is used as the initial condition for the next time step. Generally, the previous solution proves to be a good initial guess for the next time step.

5-1 Transcription methods

Transcription methods for optimal control can be categorised in different groups. First off, there are direct and indirect methods. In direct methods, the problem is first discretised and then optimised. In indirect methods, the necessary conditions for optimality are calculated first. Then, these conditions are discretised and solved. The transcription methods considered in this thesis, rely on direct transcription. Indirect transcription methods are very accurate, but hard to solve in practice as they require very accurate initial guesses to converge [33].

Direct methods can be divided in simultaneous and sequential approaches. A single shooting method is an example of a sequential method. In this method, only the control trajectories are discretised, by a piecewise smooth approximation. The discretised control trajectories then enter the NLP. A single shooting method can be compared to a cannon shooting at a target. First, an estimate is made of a good shooting angle. Then it is checked whether the target has been hit. If not, the angle is adjusted based on the previous result so that the

problem is solved sequentially using multiple iterations. The resulting NLP using a single shooting method is generally a smaller sized problem, but highly nonlinear [34].

Convergence can be improved by considering a Multiple shooting method, where the problem size is bigger, but includes less nonlinearities. Compared to a single shooting method, a multiple shooting method typically results in better convergence of NLPs and a higher degree of numerical stability (i.e. less sensitive to errors due to e.g. poorly chosen initial guesses) [35]. Furthermore, the resulting NLP has a sparser structure that can be exploited [34]. In a multiple shooting method, not only the control trajectories are discretised, but also the state trajectories. A direct collocation method results in an even larger NLP, but yields an even sparser structure compared to multiple shooting. Both the multiple shooting method and direct collocation method are explained in more detail in the next subsections.

The decision variables in this chapter are hereafter denoted by x.

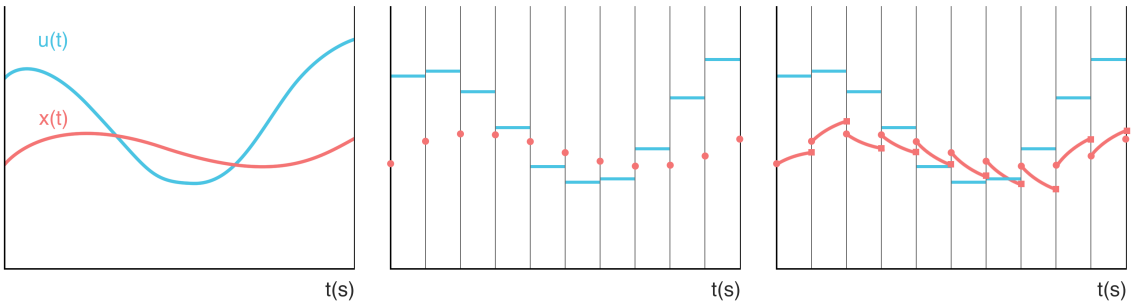
5-1-1 Multiple Shooting method

A shooting method is a transcription method that is based on simulation. Typically, a Runga-Kutta scheme is used to obtain the discrete-time dynamics. In Multiple shooting methods only the states and control inputs at the beginning of each control interval are included. There are no intermediate collocation points as seen in the direct collocation method.

In figure 5-1 the multiple shooting method is shown graphically. First, the continuous (5-1a) state and control trajectories are discretised (5-1b). The amount of control intervals is denoted by N. The control inputs $u(t)$ are discretised to obtain the decision variables $u_0, u_1, \ldots, u_{N-1}$:

$$u(t) = u_k, \quad \text{for } t \in [t_k, t_{k+1}], \quad k = 0, \ldots, N-1 \tag{5-3}$$

Similarly, the state decision variables $x_0, x_1, \ldots, x_N$ are are defined. Now, every interval k consists of a control signal u_k and a start state x_k, as shown in figure 5-1b. Starting from x_k and by integrating over each interval, the state at the next interval x_{k+1} can be determined.



(a) Continuous state and control **(b)** Discretised state and control **(c)** Integrated state trajectories

Figure 5-1: Multiple shooting method

In figure 5-1c it can be seen that there is a mismatch between the state obtained by integrating and the start state of the next time step. Continuity constraints are added to make sure that there is no mismatch between these two states:

$$x_{k+1} - \tilde{x}_{k+1} = 0 \tag{5-4}$$

In this thesis, a Runge-Kutta approach is used to obtain the state at the next time step x_{k+1}. Runge-Kutta methods are a class of methods to integrate ordinary differential equations. In this case, a fourth order Runge-Kutta method is employed. It approximates the solution to a differential equation $\dot{x} = f(t, x)$ with initial condition $x(t_0) = x_0$. The order of the Runge-Kutta method refers to the amount of approximations of the slope within one time step. In figure 5-2 a visualisation for the fourth order Runge-Kutta method can be observed. The slope at the start point and end point are defined by R_1 and R_4. The slopes at the two midpoints are R_2 and R_3.

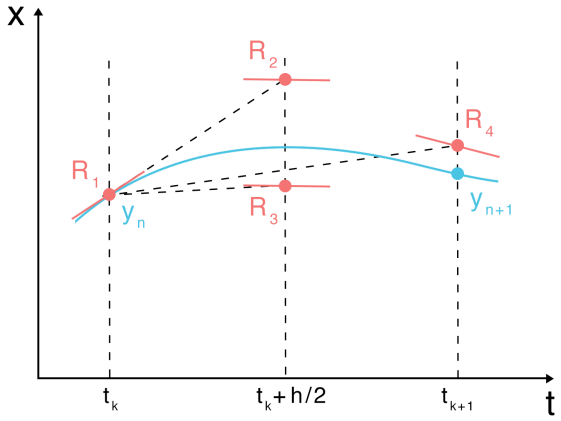


Figure 5-2: Fourth order Runge-Kutta method

To determine an approximation of x_{k+1}, a weighted average of the slopes at R_1, R_2, R_3 and R_4 is used:

$$\begin{aligned} \tilde{x}_{k+1} &= x_k + \frac{1}{6}(R_1 + 2R_2 + 2R_3 + R_4)h \\ t_{k+1} &= t_k + h \end{aligned} \tag{5-5}$$

The midpoint slopes are weighted twice as much as the start and end point. The individual slopes at each point are defined as:

$$\begin{aligned} R_1 &= f(t_k, x_k) \\ R_2 &= f(t_k + h/2, x_k + R_1/2) \\ R_3 &= f(t_k + h/2, x_k + R_2/2) \\ R_4 &= f(t_k + h, x_k + R_3) \end{aligned} \tag{5-6}$$

5-1-2 Direct collocation method

In a direct collocation method, the states and control inputs not only enter the NLP at the beginning of each control interval (as in multiple shooting), but also at intermediate collocation points. Similar to Multiple Shooting, the control trajectory is discretised into control intervals [34]:

$$u(t) = u_k, \quad \text{for } t \in [t_k, t_{k+1}], \quad k = 0, \ldots, N-1 \tag{5-7}$$

Here, N denotes the total amount of control intervals within the horizon. The states are then discretised at these discretisation points as well as at intermediate collocation points. The collocation points are placed at the so-called Legendre time points of order $d = 3$:

$$\tau = [0 \ 0.112702 \ 0.500000 \ 0.887298] \tag{5-8}$$

The discretised control intervals and collocation time points define all the points in time at which the states are evaluated:

$$t_{k,j} = t_k + h_k \tau_j, \quad \text{for } k = 0, \ldots, N-1 \text{ and } j = 0, \ldots, d \tag{5-9}$$

And the final time is defined as: $t_{N,0}$. $x_{k,j}$ is defined as the states at the individual time points. h_k denotes the time in between two steps: $h_k = t_{k+1} - t_k$.

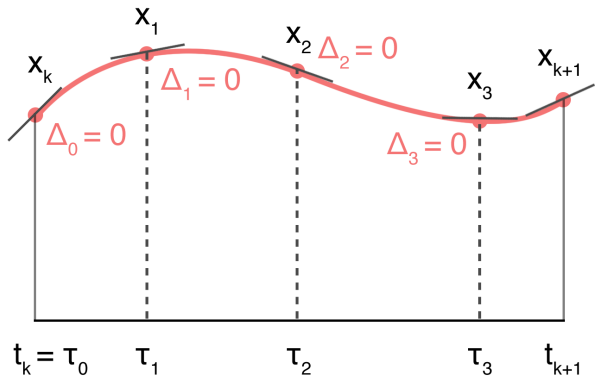


Figure 5-3: Direct collocation method for one control interval

In figure 5-3 the interpolated collocation points for one control interval are shown graphically. For the Multiple Shooting method, only the states at the beginning of each interval were considered (figure 5-1). In the direct collocation method shown here, three collocation points are added within each control interval. This way, a more accurate description of the continuous time dynamics is obtained.

For each control interval a Lagrangian basis polynomial $\ell_j(\tau)$ is defined to determine the interpolated collocation points [34]:

$$\ell_j(\tau) = \prod_{m=0, m\neq j}^{d} \frac{\tau - \tau_m}{\tau_j - \tau_m} \tag{5-10}$$

$$\ell_j(\tau) = \prod_{m=0, m\neq j}^{d} \frac{\tau - \tau_m}{\tau_j - \tau_m} = \frac{\tau - \tau_0}{\tau_j - \tau_0} \cdots \frac{\tau - \tau_{j-1}}{\tau_m - \tau_{j-1}} \frac{\tau - \tau_{j+1}}{\tau_m - \tau_{j+1}} \cdots \frac{\tau - \tau_{d-1}}{\tau_m - \tau_{d-1}} \tag{5-11}$$

$$\ell_j(\tau_m) = \begin{cases} 1, & \text{if } j = m \\ 0, & \text{otherwise} \end{cases} \tag{5-12}$$

The states can then be approximated as a linear combination of the Lagrange basis polynomials:

$$\tilde{x}_k(t) = \sum_{m=0}^{d} \ell_m\left(\frac{t - t_k}{h_k}\right) x_{k,m} \tag{5-13}$$

The approximations of the state derivatives at the individual collocations points (except at τ_0) are:

$$\tilde{\dot{x}}_k(t_{k,j}) = \frac{1}{h_k} \sum_{m=0}^{d} \dot{\ell}_m(\tau_j) x_{k,m} \tag{5-14}$$

The state at the end of a control interval is defined as:

$$\tilde{x}_{k+1,0} = \sum_{m=0}^{d} \ell_m(1) x_{k,m} \tag{5-15}$$

To ensure that the state derivatives are approximated correctly the so-called collocation equations enter the NLP as a constraint (as shown in figure 5-3):

$$\Delta_{k,j} = h_k f(t_{k,j}, x_{k,j}, u_k) - \sum_{m=0}^{d} \dot{\ell}_m(\tau_j) x_{k,m} = 0, \quad \text{for } k = 0, \ldots, N-1 \text{ and } j = 1, \ldots, d \tag{5-16}$$

Furthermore, the approximated end state enters the NLP as well to ensure that the states at the next time step are equal to the state at the end of the previous time step:

$$x_{k+1,0} - \sum_{m=0}^{d} \ell_m(1) x_{k,m} = 0, \quad k = 0, \ldots, N-1 \tag{5-17}$$

5-2 Sequential Quadratic Programming

Now that both transcription methods have been defined, NLP methods are considered to solve the posed optimisation problem.

In SQP an NLP is modelled as a Quadratic Programming (QP) subproblem at each iteration x_k. This subproblem is then solved to find the next iteration x_{k+1} [36]. This process is iterated to find an optimal solution x^*. In contrast to to IPM, in SQP the iterations do not require to be feasible. The NLP solver that is used is qpOASES [30]. It incorporates two established algorithms in nonlinear optimisation: The Active Set method and Newton's method.

SQP aims to solve the following general optimisation problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{5-18}$$

ACADO allows for condensed formulation of the system dynamics to decrease the computational complexity. In this condensed formulation the states of the decision variables are eliminated and evaluated as a function of the current time states and future time control inputs [37].

In SQP the next iteration is found by solving a quadratic subproblem of the form [36]:

$$\begin{aligned} \min_{d_k^x} \quad & (r_k)^T d_k^x + \frac{1}{2}(d_k^x)^T B_k d_k^x \\ \text{subject to} \quad & \nabla h(x_k)^T d_k^x + h(x_k) = 0 \\ & \nabla g(x_k)^T d_k^x + g(x_k) \leq 0 \end{aligned} \tag{5-19}$$

.

Here, $d_x = x - x_k$. Now, r_k and B_k need to be chosen such that the subproblem reflects the original problem. This can be done by taking the local quadratic approximation of f at x_k, resulting in setting r_k to be equal to the gradient of f at point x_k. B_k is chosen as the Hessian of f at point x_k, $\nabla^2_{xx}\mathcal{L}(x_k, \lambda_k, \nu_k)$, but is approximated by a Gauss-Newton Hessian approximation. For an objective of $f(x) = ||r(x)||_2^2$ the Gauss-Newton Hessian becomes [38]:

$$B_k = 2\nabla_x r(x_k) \nabla_x r(x_k)^T \tag{5-20}$$

A Lagrangian function incorporates all information of the optimisation problem in 5-18 into one function:

$$\mathcal{L}(x, \lambda, \nu) = f(x) + \lambda^T h(x) + \nu^T g(x) \tag{5-21}$$

Here, $\lambda \in \mathbb{R}^p$ and $\nu \in \mathbb{R}^m$ are the Lagrange multipliers for the equality and inequality constraints, respectively. The Lagrangian multipliers denote the change in the objective

function with respect to a change in the respective constraints. SQP uses a quadratic model of the Lagrangian function as the objective to be minimised, such that the nonlinearities of the original problem are captured [36]:

$$\begin{aligned} \min_{x} \quad & \mathcal{L}(x, \lambda^*, \nu^*) \\ \text{subject to} \quad & h(x) = 0 \\ & g(x) \leq 0 \end{aligned} \tag{5-22}$$

where λ^* and v^* denote the optimal multipliers. As the optimal multipliers are unknown, approximations λ_k and v_k can be used instead. The quadratic Taylor-series approximation of the Lagrangian in x is used as the objective function instead of the original objective function [36]. The advantage of this function is that its iterations are identical to iterations generated using Newton's method. The quadratic Taylor-series approximation of the Lagrangian is:

$$\mathcal{L}(x_k, \lambda_k, \nu_k) + \nabla\mathcal{L}(x_k, \lambda_k, \nu_k)^T d_k^x + \frac{1}{2}(d_k^x)^T H\mathcal{L}(x_k, \lambda_k, \nu_k) d_k^x \tag{5-23}$$

The gradient of the Lagrangian forms the following system and is equivalent to the Karush-Kuhn-Tucker Conditions (KKT) conditions:

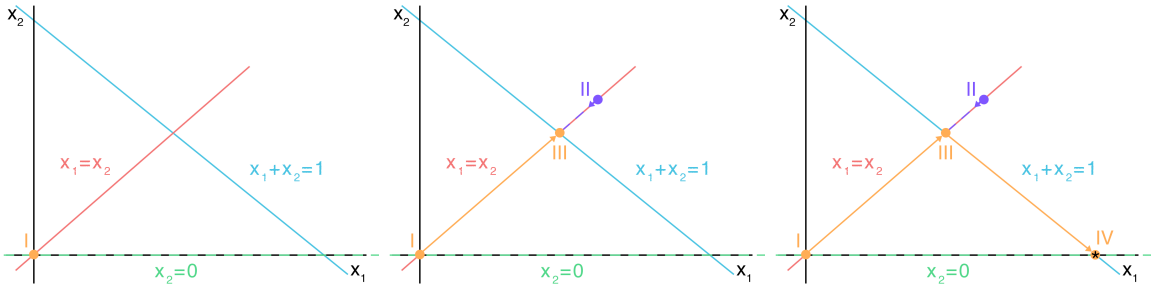
$$\nabla\mathcal{L}(x, \lambda, \nu) = \begin{bmatrix} \frac{dL}{dx} \\ \frac{dL}{d\lambda} \\ \frac{dL}{d\nu} \end{bmatrix} = \begin{bmatrix} \nabla f + \lambda\nabla h + \nu\nabla g^* \\ h \\ g^* \end{bmatrix} \tag{5-24}$$

As mentioned previously the first term contains the optimisation problem of (5-18), which is minimised by regarding its derivative and setting this term equal to zero. The second KKT condition is there to ensure feasibility, as $h(x) = 0$ is a constraint imposed in the original optimisation problem. In the third KKT condition, the active set is denoted by g^*. The KKT conditions need to be met for the solution to be considered optimal [39]. As SQP incorporates an active set strategy only active inequality constraints are taken into account for the optimisation. This means that inequality constraints far away from the optimal solution are temporarily ignored and are called inactive. When an inequality constraint is violated, this inequality constraint becomes active and may be treated as an equality constraint. Using backtracking, the step length α is chosen such that solution is on the boundary of the feasible region again.

In figure 5-4 the procedure for a simple example of the active set method is shown. In most cases, the optimal solution lies on the boundary of the feasible set, as in the example shown below. In the case where the optimal solution lies within the feasible set, all Lagrangian multipliers disappear and the problem will be solved as if it is unconstrained.

At the initial point $x_0 = (0, 0)$ (I) the green ($-x_1 + x_2 \leq 0$) and red ($x_2 \geq 0$) inequality constraints become active. The blue constraint ($x_1 + x_2 \leq 1$) remains inactive at the initial point. When the step length is too large, the next iteration could end up outside of the feasible set (II). Therefore the step length needs to be chosen such that it is as large as possible but the next iteration remains within the feasible set (III). This can be achieved by means of

backtracking. In point III the blue and red constraint become active and the green constraint becomes inactive. Then it is checked whether the Lagrange multiplier for both constraints are positive. When the Lagrange multiplier is negative, the constraint associated with that multiplier is dropped and becomes inactive. In this case, the Lagrange multiplier for the red constrained is negative, resulting in this constrained being dropped. When the Lagrange multipliers for all active constraints are positive, the associated point is the optimal solution. The latter is the case for point IV and therefore an optimal solution is found here.



(a) Initial point **(b)** Backtracking **(c)** Optimal solution

Figure 5-4: Optimal solution

The Hessian of the Lagrangian is defined as:

$$H\mathcal{L}(x,\lambda,\mu) = \begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x,\lambda,\nu) & \nabla h & \nabla g \\ \nabla h & 0 & 0 \\ \nabla g & 0 & 0 \end{bmatrix} \tag{5-25}$$

The quadratic subproblem can then be rewritten as:

$$\begin{aligned} \min_{d_x} \quad & \nabla\mathcal{L}(x_k,\lambda_k,\nu_k)^T d_k^x + \frac{1}{2}(d_k^x)^T B_k d_k^x \\ \text{subject to} \quad & \nabla h(x_k)^T d_k^x + h(x_k) = 0 \\ & \nabla g(x_k)^T d_k^x + g(x_k) \leq 0 \end{aligned} \tag{5-26}$$

.

At iteration k, the NLP (5-2) can be cast as the following quadratic subproblem:

$$\begin{aligned} \min_{d_k^x} \quad & \nabla f(x_k)^T d_k^x + \frac{1}{2}(d_k^x)^T B_k d_k^x \\ \text{subject to} \quad & \nabla h(x_k)^T d_k^x + h(x_k) = 0 \\ & \nabla g(x_k)^T d_k^x + g(x_k) \leq 0 \end{aligned} \tag{5-27}$$

Equations (5-26) and (5-27) are equivalent when there are only equality constraints. In the case of inequality constraints they are equivalent if the NLP is posed using slack variables s:

$$\begin{aligned} \min_{x\in\mathbb{R}^n, d\in\mathbb{R}^p} \quad & f(x) \\ \text{subject to} \quad & h(x) = 0 \\ & g(x) + s = 0 \\ & s \geq 0 \end{aligned} \tag{5-28}$$

The solution d_x found by solving the QP in (5-27) is then used to obtain a new iterate x^{k+1}. Aside from defining x^{k+1}, new Lagrangian multipliers λ^{k+1} and ν^{k+1} are needed for the next iteration. The directions for the multipliers are defined as:

$$\begin{aligned} d_k^\lambda &= \lambda_{qp} - \lambda_k \\ d_k^v &= v_{qp} - v_k \end{aligned} \tag{5-29}$$

Where λ_{qp} and ν_{qp} are the optimal Langange multipliers of the QP in (5-27). Then, a steplength parameter α needs to be chosen such that:

$$\phi(x_k + \alpha d_k^x) \leq \phi(x_k) \tag{5-30}$$

Here, ϕ is a merit function. The purpose of the merit function is to track and measure the progress of the optimisation, such that global convergence can be guaranteed. When the value of the merit function ϕ reduces, a step towards a solution has been taken. When the inequality in (5-30) holds for a time step k, the value of ϕ will be reduced for the next time step. The step length α needs to be adjusted at each time step to guarantee a reduction of ϕ at each time step.

The next iterates for x, λ and ν can then be written as:

$$\begin{aligned} x_{k+1} &= x_k + \alpha d_k^x \\ \lambda_{k+1} &= \lambda_k + \alpha d_k^\lambda \\ \nu_{k+1} &= \nu_k + \alpha d_k^\nu \end{aligned} \tag{5-31}$$

When d_k^x is smaller than a set relative tolerance δ and the KKT conditions are met, the solution has converged.

5-3 Interior-point Method

Aside from SQP, an Interior-Point Method (IPM) is used to solve the NLP shown in (5-2). The NLP solver that is used is IPOPT [28]. It incorporates a so-called Primal-Dual Barrier approach. It allows to solve problems of the form:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ \text{subject to} \quad & c(x) = 0 \\ & x \geq 0 \end{aligned} \tag{5-32}$$

It also possible to define a lower bound for x here, but this is omitted for the sake of simplicity. A barrier method typically moves an inequality constraint to the cost function. It does so by including a logarithmic barrier term in the cost function:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & \phi_\mu(x) = f(x) - \mu \sum_{i=1}^{n} \ln(x_i) \\ \text{subject to} \quad & c(x) = 0 \end{aligned} \tag{5-33}$$

The parameter μ is used to penalise states being close to the barrier. When μ is decreased iteratively, the minimum is reached. This problem is equivalent to solving the primal-dual equations:

$$\begin{aligned} \nabla f(x) + \nabla c(x)\lambda - \nu &= 0 \\ c(x) &= 0 \\ XNe - \mu e &= 0 \end{aligned} \tag{5-34}$$

The primal-dual equations of (5-34) together with $x, \nu \geq 0$ denote the Karush-Kuhn-Tucker (KKT) conditions. X and N represent diagonal matrices with the indices of x and ν, respectively.

The first equation in (5-34) is the gradient of the Lagrangian function (5-35). The Lagrangian allows to combine all constraints and the function to be minimised in one function. Evaluating where its gradient equals zero will therefore yield the minimum.

$$\mathcal{L}(x, \mu, z) = f(x) + c(x)^T \lambda - \nu \tag{5-35}$$

Here, $\lambda \in \mathbb{R}^m$ is a Lagrange multiplier that refers to the equality constraint $c(x) = 0$ in (5-32). The inequality constraint $x \geq 0$ in (5-32) is incorporated by the Lagrange multiplier $\nu \in \mathbb{R}^n$. To speed up the iterative process of decreasing μ, Newton steps can be taken to approach the optimum. The search directions d_k^x, d_k^λ and d_k^ν need to be found to find the next iteration $x_{k+1}, \lambda_{k+1}, \nu_{k+1}$:

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k d_k^x \\ \lambda_{k+1} &= \lambda_k + \alpha_k d_k^\lambda \\ \nu_{k+1} &= \nu_k + \alpha_k d_k^\nu \end{aligned} \tag{5-36}$$

Here, α_k denotes the step size. The search directions are found by solving equation (5-37) for d_k^x, d_k^λ and d_k^ν:

$$\begin{bmatrix} W_k & A_k & -I \\ A_k^T & 0 & 0 \\ N_k & 0 & X_k \end{bmatrix} \begin{bmatrix} d_k^x \\ d_k^\lambda \\ d_k^\nu \end{bmatrix} = - \begin{bmatrix} \nabla f(x_k) + A_k \lambda_k - \nu_k \\ c(x_k) \\ X_k N_k e - \mu_j e \end{bmatrix} \tag{5-37}$$

Here, $A_k = \nabla c(x_k)$ and W_k is Hessian of the Lagrangian (5-35):

$$W_k = \nabla^2_{xx} \mathcal{L}(x_k, \lambda_k, \nu_k) = \nabla^2_{xx} (f(x_k) + c(x_k)^T \lambda_k - \nu_k) \tag{5-38}$$

N_k and X_k are defined as:

$$N_k = \begin{bmatrix} \nu_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \nu_n \end{bmatrix}, \quad X_k = \begin{bmatrix} x_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & x_n \end{bmatrix} \tag{5-39}$$

The nonsymmetric linear system (5-37) can be solved equivalently by solving a symmetric linear system:

$$\begin{bmatrix} W_k + \Sigma_k & A_k \\ A_k^T & 0 \end{bmatrix} \begin{bmatrix} d_k^x \\ d_k^\lambda \end{bmatrix} = - \begin{bmatrix} \nabla \phi_{\mu_j}(x_k) + A_k \lambda_k \\ c(x_k) \end{bmatrix} \tag{5-40}$$

Where $\Sigma_k = X_k^{-1} N_k$. After having solved (5-40) for d_k^x and d_k^λ, the search direction d_k^z can be expressed as:

$$d_k^z = \mu_j X_k^{-1} e - \nu_k - \Sigma_k d_k^x \tag{5-41}$$

Chapter 6

Simulation results

The designed MPC is tested in IPG CarMaker 6.0.4 for Simulink, using a Volvo XC60 vehicle model. Simulations are carried out on a HP EliteBook with an Intel Core i7 processor. During simulation runs, it is assumed that all states can be measured completely without any measurement errors or disturbances. Both an Interior-Point Method (IPM) and Sequential Quadratic Programming (SQP) approach are considered in this chapter.

Figure 6-1: Volvo XC60 in CarMaker

Parameter	Value
m	2316.5 kg
l_f	1.3590 m
l_r	1.5060 m
I_z	3921.25 kg m^2
w	1.659 m

Table 6-1: XC60 vehicle parameters

In table 6-1 the vehicle parameters for a Volvo XC60 are shown. These parameters are used in conjunction with the vehicle model from chapter 3. In all simulation runs in this chapter, the initial velocity is set to $v_{x_0} = 20\,\mathrm{m/s}$. The Magic Formula tyre parameters are determined as in [40] and can be observed in table 6-2. The road friction coefficient is set to $A = \mu = 1$.

It must be noted that drawing a fair comparison between both methods is notoriously difficult. Computation times for both methods differ greatly, which results in different rates at which the MPC can be sampled. For different sampling times, different cost functions need to be considered, resulting in a difference in performance. In sections 6-2 and 6-1, the main focus is on achieving high lateral accelerations and short longitudinal displacements. This way, the tyres are more likely to operate in the nonlinear region, which is the main motivation for employing Nonlinear Model Predictive Control (NMPC). By considering an aggressive

Parameter	Value
A	1
B	$22.5554 - 0.0016F_{z_i}$
C	1.3842
D	F_{z_i}
E	1.1304

Table 6-2: Magic Formula tyre parameters

manoeuvre, it can be assessed what the limits of NMPC are, in terms of real-time limit handling capabilities.

Additional results to the results displayed in this chapter, are shown in the appendix of this thesis. In section A-2 and A-4 additional results of the steering only scenario can be observed, for IPM and SQP respectively. In section A-3 additional results are shown for the combined steering and braking scenario, using IPM. The results of more relaxed tunings that exploit the computational limits of both solvers are shown in section A-1. This section aims at providing a balanced comparison of both methods, as much as possible. In the appendix, the computation times are shown for each manoeuvre. Moreover, it is assessed whether the nonlinear part of tyre is utilised for each tyre (in the appendix). An overview of the locations of all results in this thesis can also be seen in table 6-7.

Throughout all simulation runs, the steering rate and steering angle are constrained by:

$$\begin{aligned} -1\,\text{rad} \leq \delta \leq 1\,\text{rad} \\ -1\,\text{rad/s} \leq \dot{\delta} \leq 1\,\text{rad/s} \end{aligned} \tag{6-1}$$

As mentioned in section 4-4, using an excessive amount of constraints is not desirable. Therefore, only the control input and the front steering wheel angle are constrained in the simulation runs considered in this section. It was found that constraints on slip or Y make for a significant increase in computation times or feasibility issues, without improving performance in any way. Therefore, the amount of imposed constraints is limited to the actuator limits only. The constraints shown in (6-1) aim to reflect the steering actuator limits.

The evaluated displacements for each approach are shown in figure 6-2. Here, X_s denotes the longitudinal displacement that is needed to reach $Y = 2\,\text{m}$. Y_{max} and Y_{min} denote the respective maximum and minimum obtained global Y position. The overshoot and undershoot are defined by $Y_{max} - 2\,\text{m}$ and $2\,\text{m} - Y_{min}$, respectively.

It must be noted that the desired displacements are specific to different scenarios. A trade-off needs to be made between obtaining good values values for X_s, Y_{min} and Y_{max}. For scenarios with oncoming traffic on the left lane, less overshoot is desired. In this case, X_s may need to be a bit longer. In emergency cases without oncoming traffic, a more aggressive response with a low X_s displacement may desired. In sections 6-1 and 6-2, the main objective is to obtain an aggressive response, with a low X_s value. For aggressive manoeuvres, the tyres operate at higher slip angles and therefore in the nonlinear region of the tyre. This gives the incentive to employ NMPC as opposed to linearised MPC.

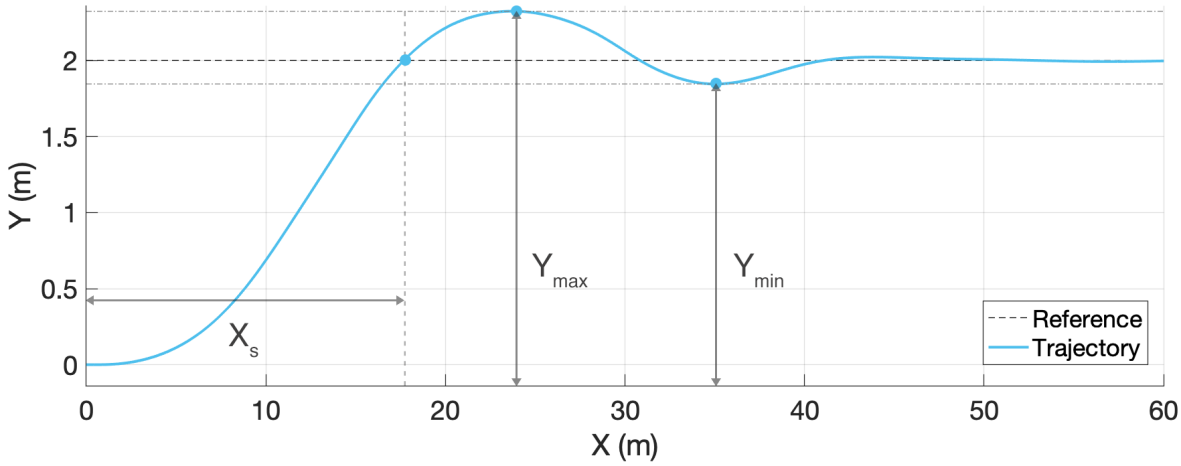


Figure 6-2: Trajectory layout

For all scenarios, the body slip $\beta = \text{atan}(v_y/v_x)$ is evaluated. This is the slip measured at the centre of gravity of the vehicle.

6-1 IPM

6-1-1 Steering only (IPM) formulation

The cost function used in this section is shown in (6-2). In CasADi, it is not possible to implement a terminal cost term, so no extra weight can be put at the states at the end of the horizon, as is possible using ACADO.

$$J = \sum_{k=1}^{N} 2(Y(t+k) - Y_r(t+k))^2 + u(t+k)^2 \tag{6-2}$$

This cost function aims at minimising the difference between reference $Y_r = 2\,\text{m}$ and the measured and predicted Y state. To ensure that no excessive steering is used, the control input is also weighted over the horizon.

The MPC time parameters for this simulation can be observed in table 6-3. They are tuned to yield an aggressive manoeuvre, with high lateral accelerations and a short longitudinal distance needed to reach the reference displacement $Y = 2\,\text{m}$.

Table 6-3: MPC parameters

Parameter	Value
N	20
T_s	0.2s
H	2s
h	0.1s

It can be seen that the sampling time T_s is relatively high at $0.2\,\text{s}$. It was found that this sampling time allowed for real-time application, as the computation times remain below the sampling time throughout the manoeuvre.

6-1-2 Steering & braking (IPM) formulation

In this section, the extended vehicle model of section 3-2-2 is used. The time parameters in this scenario remain the same as was shown in table 6-3. The initial velocity also remains the same at 20m/s. To incorporate the longitudinal wheel dynamics, the following parameters are defined:

Parameter	Value
R_w	$0.365\,\mathrm{m}$
B_d	$1\,\mathrm{Nms/rad}$
I_{wz}	$1.772\,\mathrm{kg\,m^2}$

Table 6-4: Longitudinal wheel parameters

For this problem a different cost function is used:

$$J = \sum_{k=1}^{N} 5(Y(t+k) - Y_r(t+k))^2 + 10^{-3}\dot{x}(t+k)^2 + u(t+k)^T R u(t+k) \tag{6-3}$$

where the input weight matrix R is defined as:

$$R = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 10^{-6} & 0 & 0 & 0 \\ 0 & 0 & 10^{-6} & 0 & 0 \\ 0 & 0 & 0 & 10^{-6} & 0 \\ 0 & 0 & 0 & 0 & 10^{-6} \end{bmatrix} \tag{6-4}$$

To force the MPC to use the brakes more, the longitudinal velocity is incorporated in the cost function. Without this term, the MPC is not inclined to use the brakes. Braking slows down the manoeuvre so it will take longer to reach the destination, thus increasing the cost over time. Therefore, without this velocity term, the brakes are barely used. The cost function shown here, was found to provide the maximum usage of the brakes, whilst still maintaining stability.

6-1-3 Trajectory (IPM)

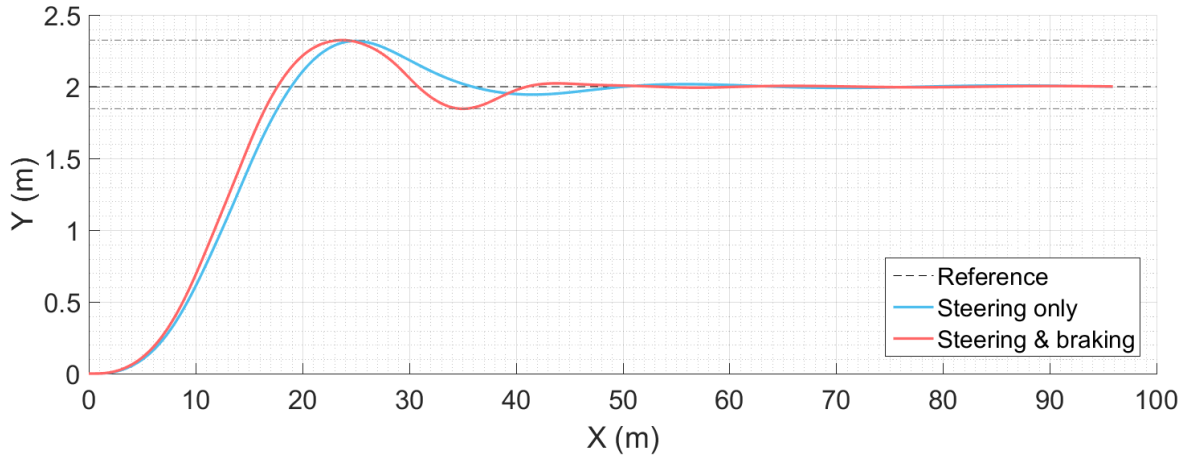


Figure 6-3: Steering & braking trajectory using IPM

The trajectories for both steering only and combined braking and steering can be observed in figure 6-3.

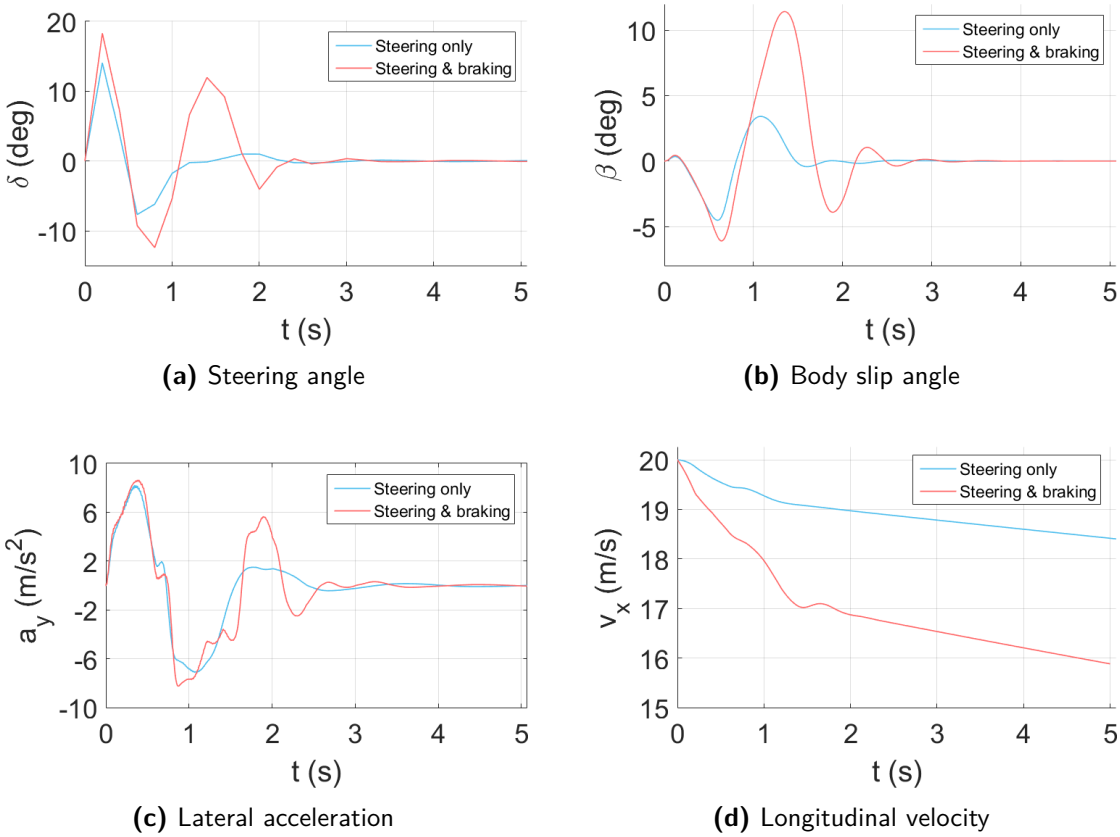
Steering only

For the steering only scenario, the overshoot is 0.32 m and there is a slight undershoot present of 0.06 m. The longitudinal distance (X_s) needed to be in the safe zone ($Y = 2$ m) is 18.96 m. A trade-off needs to be made between obtaining a low overshoot and a quick response. As the goal here is to avoid a vehicle, it is desired to be in the safe zone as quickly as possible. This comes at the cost of a slight overshoot. In case there is another vehicle in the lane to the left, the overshoot might be more problematic. This goes to show that for different scenarios, different trajectories are required. In this thesis, the focus is on an aggressive manoeuvre, so a slight overshoot is deemed acceptable. The mean computation time for this scenario is $t_{c,mean} = 0.136$ s. This is the average time it takes to pose and solve the Nonlinear Programme (NLP), for each time step.

Steering & braking

For the combined steering and braking scenario, the most notable difference compared to the steering only scenario is a slightly larger undershoot. The undershoot here reaches as far as 0.15 m below the reference line. The overshoot here is slightly larger as well, at 0.32 m. It takes 17.67 m to be in the safe zone, which is 1.27 m lower than the scenario with steering only. The increased undershoot may be undesirable in case an obstacle is present at around 35 m. Although a more complex MPC problem is posed compared to the steering only scenario, the mean computation time increases only slightly to $t_{c,mean} = 0.170$ s.

6-1-4 Notable results (IPM)



(a) Steering angle

(b) Body slip angle

(c) Lateral acceleration

(d) Longitudinal velocity

Figure 6-4: Results of steering & braking scenario using IPM

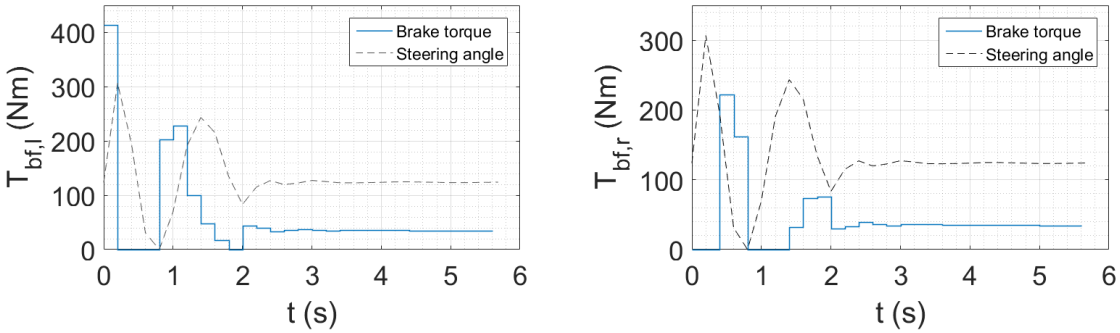
Steering only

The aggressiveness of the manoeuvre is apparent from the steering angle plot (6-4a), especially in the first segment where the car makes a left turn. The body slip is displayed in figure 6-4b. This is the slip angle, measured at the centre of gravity of the vehicle. The two bursts of slip are the results of the two short consecutive turns. The lateral acceleration in figure 6-4c reaches a maximum of $8.19\,\mathrm{m/s^2}$ and a minimum of $-7.10\,\mathrm{m/s^2}$. At the end of the manoeuvre the longitudinal velocity v_x (6-4d) is $18.42\,\mathrm{m/s}$.

Steering & braking

For the steering and braking scenario, the MPC tries to correct for the undershoot at around $X = 35\mathrm{m}$. This can be seen from the steering angle output (6-4a) around the 1.4s mark, where the steering angle is much larger than in the steering only scenario. Much more slip (6-4b) is obtained is during the manoeuvre, especially during the right turn of the manoeuvre. The maximum value for a_y is $8.64\,\mathrm{m/s^2}$. Due to increasing steering angle at the $1.4\,\mathrm{s}$ mark, a higher lateral acceleration (6-4c) is obtained (at $2\,\mathrm{s}$) as the MPC is trying to correct for

the undershoot. After 5 s the longitudinal velocity (6-4d) has decreased to 15.98 m/s, which is 2.44 m/s lower than the scenario with steering only.



(a) Front left wheel **(b)** Front right wheel

Figure 6-5: Brake torques at the front wheels

The brake torques at the individual wheels are shown in 6-5, along with the steering angle. The steering angle shown here is not true to size but is solely there to provide a reference of the steering direction. It can be seen that brake torque is applied at the left wheels as the vehicle is making the first left turn. When the vehicle starts turning to the right, brake torque is requested from the right wheels. Braking at the inner wheels results in extra yaw moment in the desired turning direction. The MPC tries to make the best of a slightly contradictory objective: braking while steering towards a desired position, as fast as possible.

As mentioned in 4-4, the brake torque control inputs remain unconstrained throughout the optimisation. It was found that constraining the brake torque to its realistic limitations resulted in at least a twofold increase of the computation times. The resulting torques shown in figure 6-5 are nowhere near the actual braking limits, so the performance is not effected.

6-2 SQP

6-2-1 Steering only (SQP) formulation

The steering only scenario is also evaluated using an SQP method. The MPC time parameters, shown in table 6-5 proved to yield good performance for the manoeuvre. The sampling time is chosen to be much smaller, as an SQP approach results in relatively short computation times for this problem, compared to IPM.

Table 6-5: MPC parameters

Parameter	Value
N	40
T_s	0.04 s
H	1.6 s
h	0.04 s

The cost function, used in ACADO is the following:

$$J = 100(Y(N) - Y_r(N))^2 + \sum_{k=1}^{N} 1 \cdot 10^{-5}(Y(t+k) - Y_r(t+k))^2 \tag{6-5}$$

As mentioned before, ACADO allows for a terminal cost term that penalises the difference between the state and reference state at the end of the horizon. In this case, the weight on this terminal cost term is much higher than the weight on the states over the horizon. Decreasing the weight of the terminal cost term resulted in unsatisfactory performance using SQP.

In this cost function there is no penalisation of the control input as the optimal control input calculated by the MPC was not found to be excessive. Furthermore, hard constraints ensure that the control input stays within reasonable bounds.

A shifting strategy is incorporated for the SQP method. With a shifting strategy, the predicted states are shifted one time step forward and used as an initial guess, essentially warm starting the optimisation to increase convergence rates. The downside of a shifting strategy is that a delay block needs to be implemented to prevent an algebraic loop, delaying the manoeuvre by one sampling time step T_s. The slight delay with lower computation times is favoured over no delay with higher computation times. Therefore, a shifting strategy is included.

6-2-2 Trajectory (SQP)

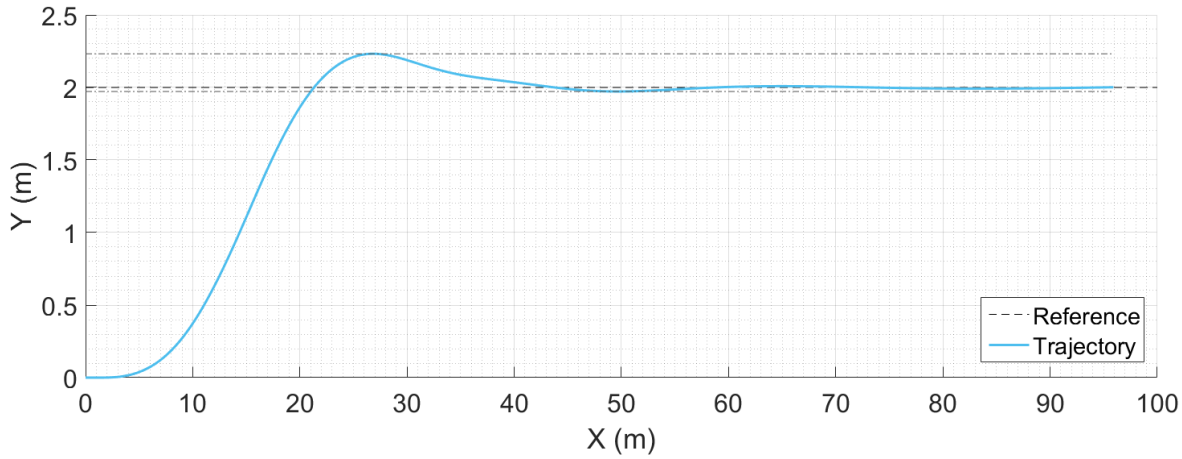


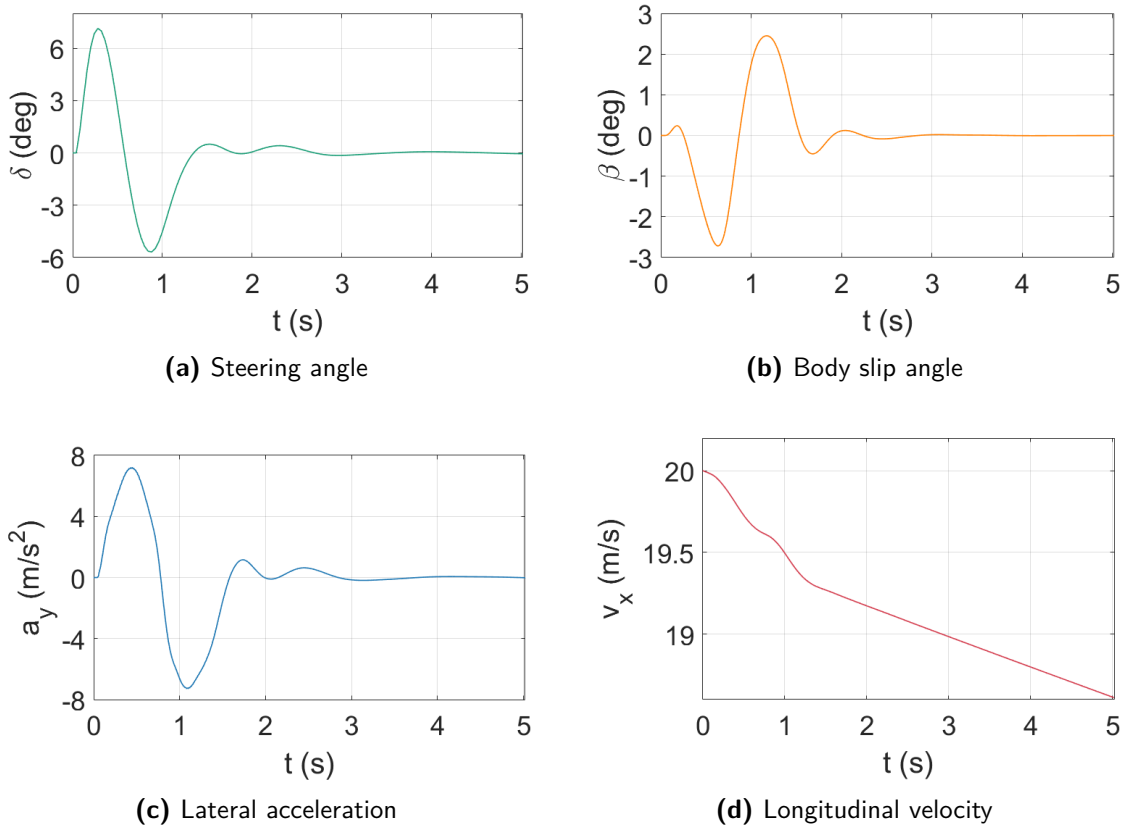
Figure 6-6: Trajectory of steering only scenario using SQP

In terms of computation times, SQP was found to be a more lightweight solution than IPM. Therefore, the MPC can run at higher sampling times. This can be ascribed to the relatively small problem size of the problem posed in the steering only scenario. If the amount of states are to increase, computation times using SQP are likely to rise quickly. The trajectory for SQP method using steering only can be observed in figure 6-6. It takes 21.32 m for the vehicle to be in the safe zone at $Y_r = 2\,\text{m}$. The overshoot is around 0.23 m and the undershoot is nearly nonexistent at 0.03 m.

6-2-3 Notable results (SQP)

Due to the lower sampling time, compared to IPM in the previous section, a much smoother steering angle is obtained, as can be seen in figure 6-7a. The amount of body slip (6-7b) is also reduced compared to IPM. It can be said that an overall smoother trajectory is obtained, which can also be noticed from the lateral acceleration in figure 6-7c. However, it must be noted that the MPC (using ACADO) is implemented by means of generated C code in an S-Function. This allows for faster computation times and thus lower sampling times than consdired in this section. In section A-1 a comparison is drawn where this is taken into account.

Overall, the SQP method proved to provide a smoother but less aggressive trajectory, compared to the IPM approach in the previous section. This is also apparent in the results shown in figure 6-7. The maximum steering angles (6-7a) and body slip angles (A-9b) are much lower. Its maximum and minimum values are 2.45 deg and -2.71 deg, respectively. The lateral acceleration (6-7c) is slightly lower, at a maximum of 7.20 m/s^2 and a minimum -7.27 m/s^2. The mean computation time is much lower than for the IPM approaches, as $t_{c,mean} = 0.0096\,\text{s}$.



(a) Steering angle

(b) Body slip angle

(c) Lateral acceleration

(d) Longitudinal velocity

Figure 6-7: Results of steering only scenario using SQP

6-3 Overview of simulation results

An overview of the results of all scenarios and tunings, considered in simulation runs, is shown in table 6-7. The main differences between CasADi and ACADO, as found in this thesis, are illustrated in table 6-6.

It was found that the IPM approach in this thesis scaled better with problem size. A combined steering and braking scenario was implemented successfully, with only a slight increase in computation times compared to a steering only scenario. The combined steering and braking scenario involved an addition of 4 states, 4 control inputs and longitudinal dynamics. However, computation times remained in the same order of magnitude.

Remarkably, the more relaxed (steering only) tuning (section A-1) for IPM yielded a lower value for X_s than the aggressive tuning in section 6-1, while a lower maximum lateral acceleration was obtained. This can be attributed to the excessive amount of slip present in the tuning found in 6-1.

For SQP, a combined steering and braking scenario could not be carried out. Compilation times for the C coded S-Function increased from around 5 minutes to around 2 hours, reducing the ease of repeatability for different tunings. In most cases, the vehicle would not converge to the desired lateral displacement at all. In other cases, the brakes were barely used ($T_{b_i} < 1\,\text{Nm}$). No tuning was found for which the combined steering and braking scenario could be

carried out successfully.

IPM proved to scale well when it comes to a change in reference displacement or change of initial velocity. For SQP this was the case as well, provided that the horizon is divided in a relatively high number of control intervals N. Furthermore, it was found that IPM converged to the desired position for a wide variety of cost functions, whereas for SQP the cost function needed to be tuned more carefully to obtain convergence.

Lastly, it was found that an SQP method performs well when it comes to heavily hard constrained optimisation problems. For IPM however, it was observed that implementing a large amount of hard constraints led to infeasibility errors or substantial increases in computation times.

	CasADi (IPM)	**ACADO (SQP)**
Large problems	+	–
Small problems	–	+
Robustness	+	∘
Scalability	+	+
Hard constraints	–	+

Table 6-6: Overview of CasADi vs. Acado

The simulation results from the different approaches considered in this thesis can be observed in table 6-7.

	Section	$a_{y,max}$	$a_{y,min}$	β_{max}	β_{min}
IPM (steering only)	6-1, A-2	$8.19\,\mathrm{m/s^2}$	$-7.10\,\mathrm{m/s^2}$	3.41 deg	−4.52 deg
SQP (steering only)	6-2, A-4	$7.19\,\mathrm{m/s^2}$	$-7.27\,\mathrm{m/s^2}$	2.45 deg	−2.71 deg
IPM (steering & braking)	6-1, A-3	$8.64\,\mathrm{m/s^2}$	$-8.23\,\mathrm{m/s^2}$	11.42 deg	−6.10 deg
IPM (steering only) II	A-1	$7.27\,\mathrm{m/s^2}$	$-7.21\,\mathrm{m/s^2}$	1.89 deg	−2.63 deg
SQP (steering only) II	A-1	$6.98\,\mathrm{m/s^2}$	$-7.28\,\mathrm{m/s^2}$	2.01 deg	−2.28 deg
	Y_{max}	Y_{min}	X_s	$v_{x,end}$	$t_{c,mean}$
IPM (steering only)	2.318 m	1.945 m	18.965 m	18.41 m/s	0.136 s
SQP (steering only)	2.232 m	1.972 m	21.322 m	18.61 m/s	0.0016 s
IPM (steering & braking)	2.324 m	1.845 m	17.691 m	15.98 m/s	0.170 s
IPM (steering only) II	2.322 m	1.925 m	18.827 m	18.03 m/s	0.147 s
SQP (steering only) II	2.260 m	1.986 m	20.426 m	18.06 m/s	0.0096 s

Table 6-7: Overview of different tunings in simulation

Chapter 7

Experimental results

7-1 Experimental setup

The test car is a Volvo 40 Cross country model equipped with a dSPACE microAutobox II system. The experimental tests shown in this chapter are carried out at Volvo's test track, Hällered Proving Ground. The initial velocity set in the MPC needs to match the real initial velocity of the vehicle. Therefore, the velocities in this chapter are given in km/h as the speedometer in the car is used to match the velocity when the function is triggered. In this chapter an experimental test is considered at an initial velocity of 70 km/h. This velocity corresponds to the velocities considered in the simulation runs. In the appendix, experimental results for 50 (B-1), 60 (B-2) and 90 (B-3) km/h are shown. The MPC performed similarly for these different velocities.

Figure 7-1: Volvo V40 test vehicle

Parameter	Value
Mass	1650 kg
l_f	1.078 m
l_r	1.570 m
I_z	2656 kg m^2
w	1.520 m

Table 7-1: V40 vehicle parameters

The dSPACE Real-time Interface (RTI) for MATLAB/Simulink is used to run the designed MPC on the embedded dSPACE system. This interface allows to build C code from Simulink models for the embedded dSPACE MicroAutoBox II system in the test vehicle. Unfortunately, MATLAB's own C code generation tools do not support external frameworks like CasADi and ACADO.

To be able to use the RTI interface, the MPC can be incorporated by means of an S-Function in Simulink. An S-Function allows to incorporate blocks written in C in Simulink. The designed MPCs, which rely on the CasADi and ACADO libraries, must be generated in C code in order to use the aforementioned S-Function block.

As of today, CasADi supports code generation for everything except for the Nonlinear Programme (NLP) solver itself. It is possible to generate code that is dependent on libraries (in this case CasADi and IPOPT). The libraries then need to be linked on the embedded system. As the dSPACE RTI only supports generated code from its MATLAB/Simulink interface (using S-Functions), the CasADi generated MPC can not be compiled for the embedded system. An alternative is to compile the CasADi core and IPOPT from source on the embedded system.

ACADO on the other hand, supports full C code generation. The generated C code for MPC can be implemented in Simulink through an S-Function block. The C code needs to be adjusted as some parts of ACADO are incompatible with dSPACE (such as internal timer related functions and print messages). After the C code is adjusted and an S-Function is built, the entire controller can be built for the embedded system using the RTI.

Due to the slightly more user friendly way of embedding C code using ACADO, only this approach is tested in the test vehicle. CasADi and ACADO are compared in simulation and it is expected that their difference in performance should be similar in physical tests. The complete code generation procedure is shown schematically in figure 7-2.

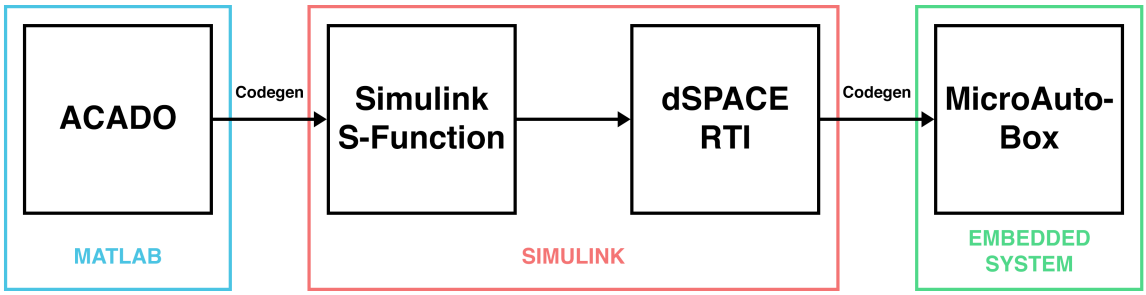


Figure 7-2: Code generation procedure

In the test scenario, the function will be triggered manually. In reality, the Evasive Manoeuvre Assistance Function (EMA) function is triggered when a set of conditions are met. As some current Volvo models are already equipped with EMA functions, these triggering conditions are already in place. During the test drives, Electronic Stability Control (ESC) was turned off.

The tyre model parameters for the test vehicle are unknown. Therefore, the same tyre model parameters are used as during simulation runs. This introduces a mismatch between the tyre model within the MPC and the real-life tyre behaviour and poses a challenge to the robustness of the MPC.

Due to the fact that timer related functions are incompatible with the used dSPACE system, they are disabled. Therefore, no computation times can considered for the experimental tests. It was found however, that a sampling frequency of 25 Hz never resulted in any task overrun errors. Task overrun errors occur when the dSPACE system is carrying out a task while the next task is already scheduled to start. In other words, the computation time is higher than the sampling time and a real-time application is not possible. When a task overrun errors occurs, the designed software need to be reflashed on the MicroAutoBox to be able to test again, which is time consuming. Due to the fact that repeatability was important to tune the

MPC properly, the sampling frequency is set at 25 Hz for all experimental tests in this thesis, even though it may be possible to run at higher frequencies in some instances.

7-2 Obtaining the measurement states

During testing, measurement signals from the on-board Inertial Measurement Unit (IMU) are used. A Controller Area Network (CAN) bus is used for communication between the actuators, the AutoBox and the sensors.

The states that are measured by the IMU are: $\dot{x}$, α_r and $\dot{\psi}$. Here, α_r denotes the slip angle measured at the rear axle. From these states, $\dot{y}$ can be obtained:

$$\dot{y} = \dot{x} \tan \alpha_r + \dot{\psi} l_r \tag{7-1}$$

$$\begin{aligned} \dot{X} &= \dot{x} \cos \psi - \dot{y} \sin \psi \\ \dot{Y} &= \dot{x} \sin \psi + \dot{y} \cos \psi \end{aligned} \tag{7-2}$$

By integrating $\dot{X}$ and $\dot{Y}$ the global positions X and Y are obtained. Measured signals are inevitably subject to noise with a slight bias resulting in so-called drift when the signal is integrated. Over longer periods of time, this will result in a significant mismatch between measured state and the true state. In order to reduce the influence of drift on the measurements, the integrators will start from a set initial condition every time the evasive manoeuvre function is triggered. This ensures that there is no build-up of the integrator's output prior to the function activation.

7-3 Torque request

During simulation runs in the previous section, it was possible to request a desired steering angle at the steering wheel. In the test vehicle, steering wheel torque is requested as opposed to steering angle. To map steering rate or angle to steering wheel torque, a steering model needs to be developed. A steering model typically relies on accurately estimated parameters such as the torsion bar stiffness and steering column damping [41].

Torque can be requested by a so-called Driver Steering Recommendation (DSR) torque request that is sent to the MicroAutoBox. DSR is in place to assist the driver in difficult handling situations by giving a slight nudge on the steering wheel to recommend the driver to turn the steering wheel. This nudge is given by a torque offset on the steering wheel in the recommended direction. Normally, the DSR torque request is limited at 3Nm for ordinary driving and at 5Nm for the autopilot function. In the V40 test vehicle these limits are turned off and the actuator is limited by 7Nm. This is the maximum amount of Torque that is possible to request through the MicroAutoBox.

A complex relation exists between wheel angle and applied torque at the steering wheels, due to the presence of a power steering module. Boost curves are used to convert driver applied torque to a total amount of torque on the pinion, which is then translated to a displacement of the rack. When the rack is displaced, the wheels turn accordingly. The assistance torque is dependent on the vehicle velocity and the driver applied torque. As advanced steering wheel modelling is not within the scope of this thesis, a simplified model is used. This provides an opportunity to test the robustness of the MPC as the model simplification makes for a mismatch between the requested and the applied steering rate.

A simplified dynamic equation for a steering model with a steering wheel applied torque T_d is used [41]:

$$J_s\ddot{\theta} = T_d - K_s\theta - b_s\dot{\theta} \tag{7-3}$$

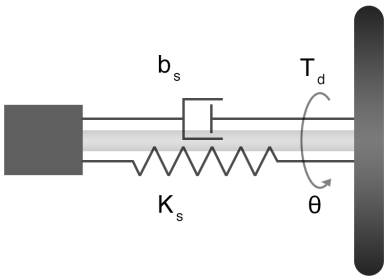


Figure 7-3: Steering wheel torque model

Where θ is the steering wheel angle. The angular velocity and acceleration of the steering wheel are denoted by $\dot{\theta}$ and $\ddot{\theta}$, respectively. The model parameters are the torsion stiffness K_s, the moment of inertia of the steering system and b_s is the damping coefficient. Furthermore, a static steering ratio ($i = 15.2$) is assumed, such that θ can be mapped directly to the front wheel angle δ.

The objective here is to map the output of the MPC to a corresponding steering wheel torque. The output of the MPC is the steering rate at the front wheels, from which a desired steering wheel rate $\dot{\theta}_{des}$ is obtained. As the MPC's output is not subject to noise, it is possible to differentiate $\dot{\theta}_{des}$ to obtain $\ddot{\theta}_{des}$.

Due to mismatches between the model used in the MPC and the car's behaviour, the measured steering wheel angle θ is used as integrating $\dot{\theta}_{des}$ leads to error build-up. This leads to the following mapping from θ_{des} to T_d [41]:

$$J_s\ddot{\theta}_{des} = T_d - K_s\theta - b_s\dot{\theta}_{des} \tag{7-4}$$

Ideally, the model in the MPC would be adjusted such that it outputs torque directly. This would introduce the need for another state so that the steering rate enters the model as a state. Unfortunately, the MicroAutoBox's computational power is limited and the addition of another state resulted in the MicroAutoBox going into deadlock. Another downside of embedding the torque model in the MPC is that it is not possible to adjust the inertia, stiffness and damping parameters on the go. Due to the complex nature of a steering model with a power assist module there is no fixed set of parameters that are correct for every speed. Therefore, it is desired that these can be adjusted easily without recompiling and flashing software on the MicroAutoBox every iteration. The dSPACE RTI allows for this. For these reasons, the torque model is outside the S-Function with the MPC and uses the MPC's output as an input to obtain desired torque.

7-4 Results

The torque model parameters in table 7-2 were found to give a reasonable mapping from steering rate to torque, with satisfactory performance. These parameters were not expected to provide a highly accurate mapping but merely tuned such that a satisfactory performance was achieved.

K_s	b_s	J_s
4.5	0.9	0.05

Table 7-2: Torque model parameters at $v_{x0} = 70\text{km/h}$

The time parameters for the experimental setup are shown in table 7-3. Electronic Control Unit (ECU)s typically operate at 100 Hz, whereas the MPC operates at 25 Hz. To cope with these different sampling times, rate transition blocks are used to convert the sampling times of the measurement signals. The measurement signals are sampled at 100 Hz and need to be converted to the MPC's sampling time, at 25 Hz. The optimal steering rate output of the MPC block is then converted back to 100 Hz, such that it can be used to request steering wheel actuation.

Parameter	**Value**
N	40
T_s	0.04 s
NT_s	1.6 s
h	0.04 s

Table 7-3: MPC time parameters for experimental setup

While the MPC parameters remain the same as in the simulation runs, the cost function is different:

$$J = 1 \cdot 10^6 (Y(N) - Y_r(N))^2 + \sum_{k=1}^{N} 5 \cdot 10^{-6} (Y(t+k) - Y_r(t+k))^2 \tag{7-5}$$

This is the same, more relaxed cost function that is used for the comparison between SQP and IPM in section A-1. It was found that this cost function makes for a more robust manoeuvre, that can also be extended to different initial velocities, as shown in the appendix.

Throughout all experimental runs, the steering rate and steering angle are constrained by:

$$\begin{aligned} -1\,\text{rad} \leq \delta \leq 1\,\text{rad} \\ -1\,\text{rad/s} \leq \dot{\delta} \leq 1\,\text{rad/s} \end{aligned} \tag{7-6}$$

7-4-1 Trajectory

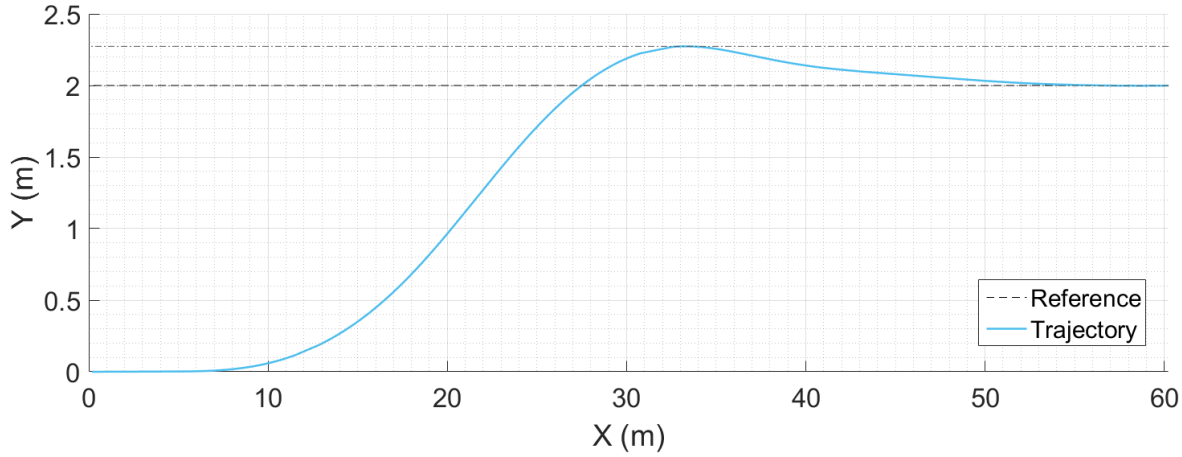


Figure 7-4: Trajectory of experimental test at $v_{x0} = 70\text{km/h}$

The trajectory for an experimental test at 70 km/h is shown in figure 7-4. It can be seen that a reasonable overshoot of 0.273m is obtained. Compared to the simulation runs, it takes much more time for the vehicle to start moving to the left (figure 7-4). This can be ascribed due to the delay between applied torque on the steering wheel and the steering angle, as shown in figure 7-5. Moreover, the torque saturates at 7 Nm and it therefore is not possible to provide more aggressive steering. This can be observed in figure 7-5. The experimental results are recorded on a rainy day, so it may be expected the real friction coefficient is lower than the one used in the tyre model within the MPC.

In figure 7-5 the requested steering wheel torque and measured steering angle are shown. It can be seen that there is a delay of around 0.13 s for the steering angle to increase as a result of the requested torque. In a real-life scenario it would be desired to account for this delay by initiating the evasive manoeuvre function slightly earlier.

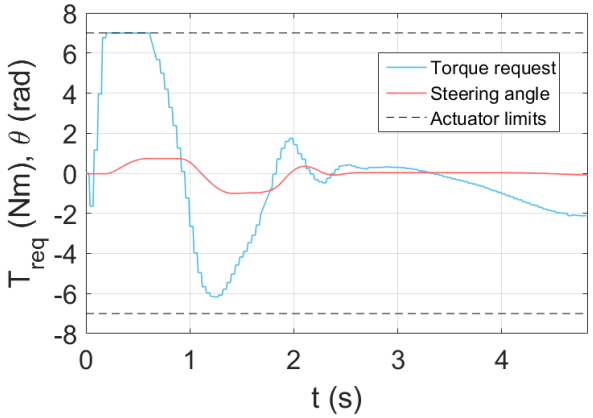
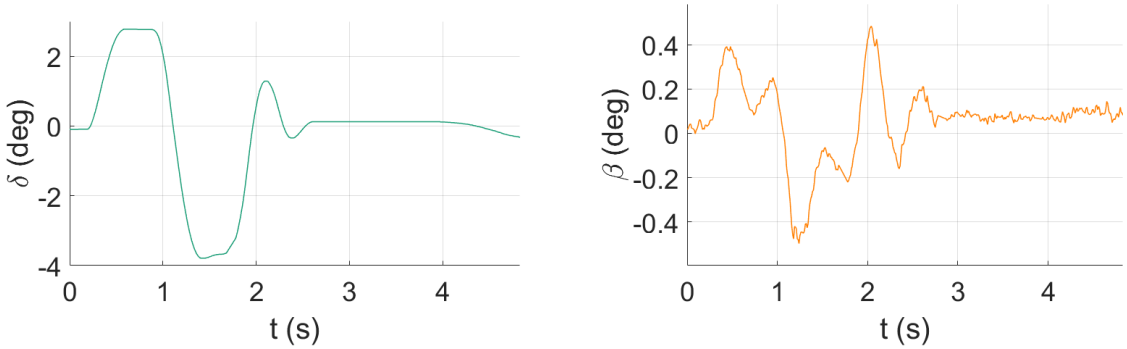


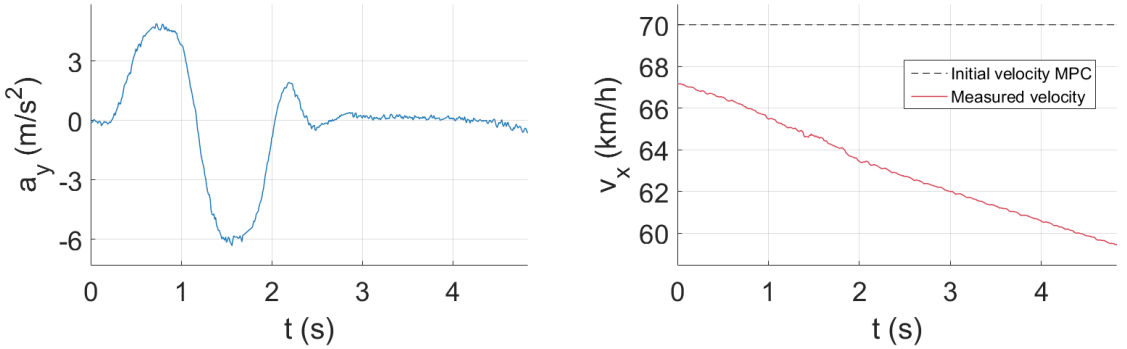
Figure 7-5: Requested torque and steering wheel angle at $v_{x0} = 70\text{km/h}$

7-4-2 Notable results

The delay between applied torque and steering angle can also be observed in figure 7-6. For practical applications, it may be desired to initiate the manoeuvre slightly earlier to account for the delay. The amount of body slip (7-6b) is lower than for the simulation runs, with a maximum of 0.50 deg. The lateral acceleration (7-6c) is also slightly less aggressive at a maximum of 6.33 $\mathrm{m/s^2}$. In figure 7-6d it can be seen that there is a slight mismatch between the initial velocity of the MPC and the velocity measured by the IMU. However, a small mismatch does not hinder performance.



(a) Steering angle **(b)** Body slip angle



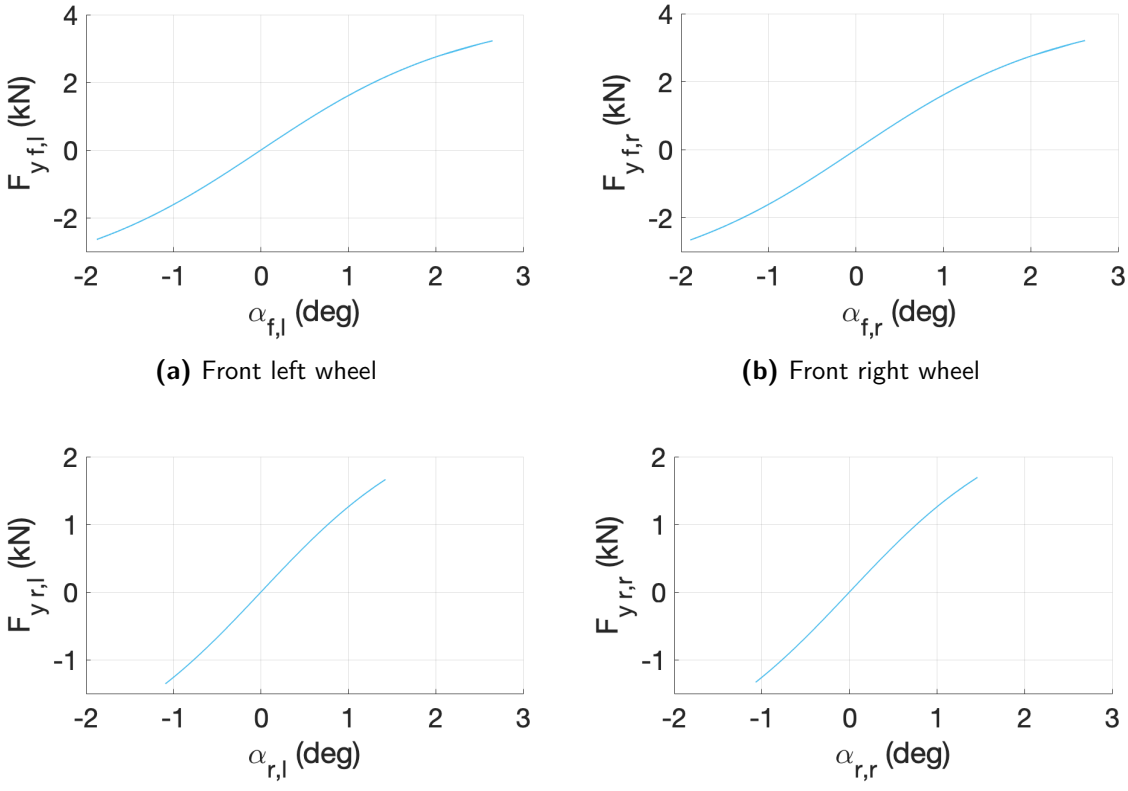
(c) Lateral acceleration **(d)** Longitudinal velocity

Figure 7-6: Results of experimental test at $v_{x0} = 70\mathrm{km/h}$

7-5 Friction utilisation

The main reason for employing nonlinear MPC, is the fact that the nonlinear part of tyre can be utilised, accurately. In figure 7-7 the tyre forces and slip angles of the individual wheels are shown. The tyre forces or slip angles can not be measured directly in the test car. Therefore, they are obtained from the steering angle, longitudinal and lateral velocities, just as the model within the MPC, shown in section 3.

It can be seen that the vehicle operates slightly in the nonlinear region for the front two tyres. As the torque exerted by the actuator is limited by 7Nm it is not possible to have the tyres operate further towards the tyre saturation limit. However, it must be noted that it can not be assumed that the tyres truly operate in the nonlinear region as the real slip angles and lateral tyre forces may differ from those obtained by the simplified magic formula in the MPC.



(a) Front left wheel

(b) Front right wheel

(c) Rear left wheel

(d) Rear right wheel

Figure 7-7: Lateral tyre forces and slip angles at $v_{x0} = 70\text{km/h}$

7-6 Overview of results

The implemented MPC strategy has shown good performance for a variety of initial velocities. With only minor tweaks to the steering wheel model parameters at different velocities, convergence to the desired lateral position could be achieved.

An overview of all experimental results is presented in table 7-4. It can be seen that more body slip is obtained at lower speeds. This can be attributed to the fact that more steering wheel displacement is needed to obtain high lateral accelerations, as compared to higher speeds. Due to the centrifugal force, lateral accelerations are higher at higher velocities. Therefore, to obtain similar lateral accelerations as at higher speeds, the steering displacement needs to larger at low speeds.

At $v_{x_0} = 50\,\text{km/h}$ and $60\,\text{km/h}$ more body slip is obtained than at $70\,\text{km/h}$, so the slip angles at the individual wheels are also higher. This means that the nonlinear region at this velocity is utilised more than as was shown in section 7-5 at $v_{x_0} = 70\,\text{km/h}$.

v_{x0}	$a_{y,max}$	$a_{y,min}$	β_{max}	β_{min}	Y_{max}	Y_{min}	X_s
$50\,\text{km/h}$	$4.08\,\text{m/s}^2$	$-5.53\,\text{m/s}^2$	$1.17\,\text{deg}$	$-1.59\,\text{deg}$	$2.125\,\text{m}$	$1.779\,\text{m}$	$21.000\,\text{m}$
$60\,\text{km/h}$	$5.03\,\text{m/s}^2$	$-6.45\,\text{m/s}^2$	$0.80\,\text{deg}$	$-0.99\,\text{deg}$	$2.267\,\text{m}$	$1.862\,\text{m}$	$23.498\,\text{m}$
$70\,\text{km/h}$	$4.87\,\text{m/s}^2$	$-6.33\,\text{m/s}^2$	$0.48\,\text{deg}$	$-0.50\,\text{deg}$	$2.273\,\text{m}$	$1.998\,\text{m}$	$27.497\,\text{m}$
$90\,\text{km/h}$	$4.79\,\text{m/s}^2$	$-5.20\,\text{m/s}^2$	$0.40\,\text{deg}$	$-0.60\,\text{deg}$	$2.390\,\text{m}$	–	$37.283\,\text{m}$

Table 7-4: Overview of all experimental results

Chapter 8

Conclusion

In this thesis a real-time Nonlinear Model Predictive Control (NMPC) strategy was developed that was able to carry out an aggressive evasive manoeuvre successfully. It was shown that the NMPC problem could be solved efficiently by regarding two state-of-the-art Nonlinear Programme (NLP) solvers.

Even though some simplifications were made in terms of vehicle and tyre models, good performance was obtained. This goes to show that the proposed MPC scheme is a robust method and a promising prospect for limit handling safety systems. By employing even more accurate vehicle models, performance is likely to increase even further. This will become more feasible as the processing power of Electronic Control Unit (ECU)s increases.

Compared to previous research, low sampling times were achieved for aggressive manoeuvres. In simulation, lateral accelerations up to $8.46\,\mathrm{m/s^2}$ were achieved. With limited actuation of the steering wheel, lateral accelerations up to $6.45\,\mathrm{m/s^2}$ were obtained for experimental tests in a test vehicle. In both cases, reasonable overshoots were obtained, while maintaining a short longitudinal distance travelled. Moreover, it was shown that the nonlinear region of the tyre was utilised, which was the main motivation for implementing NMPC. A sampling frequency of $25\,\mathrm{Hz}$ was accomplished on the embedded system in the test vehicle.

This thesis has shown that the proposed MPC strategy is able to handle model mismatches and measurement inaccuracies well. Furthermore, the computational complexity that has always been an issue with NMPC has not shown to be problematic for real-time execution. It can be concluded that a mature MPC strategy on ECUs may very well be a feasible solution for future limit handling applications.

All in all, it can be concluded that Sequential Quadratic Programming (SQP) generally performs better for smaller constrained problems, whereas an Interior-Point Method (IPM) is more suited for larger unconstrained problems.

8-1 Recommendations

8-1-1 Stability constraints

In this thesis, stability of the vehicle during the manoeuvre was not ensured. Stability behaviour of the vehicle can be evaluated by means of a nonlinear phase plot of the system dynamics. In figure 8-1 an example is shown for a nonlinear phase diagram for the vehicle model used during simulations in this thesis. By evaluating the trajectories from each initial point, a set of stable initial points can be obtained, here denoted by circles. In this example, $\delta = 0$, and $v_x = 20\,\mathrm{m/s}$. For different values of δ and v_x, different phase portraits are obtained, with different stable regions. Look-up tables can be implemented to obtain stable regions for different values of δ and v_x. The stable regions can then be passed to the optimisation, as constraints on $\dot{\psi}$ and β. This way, stability can be ensured during the manoeuvre.

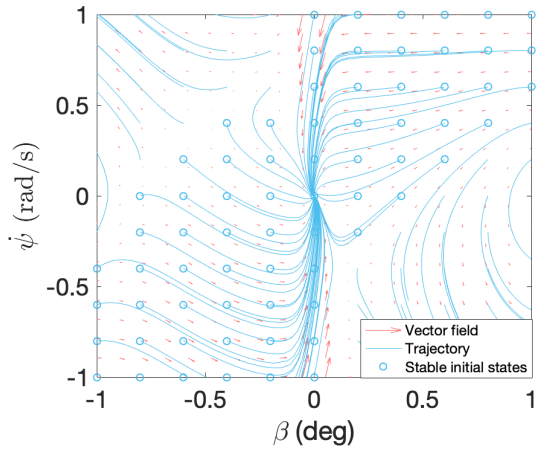


Figure 8-1: Nonlinear phase portrait

8-1-2 More advanced vehicle model

Relatively simple vehicle models and tyre models were used in this thesis. The vehicle model could be extended by implementing roll and load transfer dynamics. For vehicles with a higher centre of gravity, roll dynamics could be especially valuable. Load transfer has a significant impact on the friction that can be obtained at each wheel. To be able to accurately approximate tyre behaviour at the limits of handling, load transfer dynamics need to be included within the MPC. This, however increases the overall computational complexity of the MPC.

To be at the edge of what the tyre can handle, a highly accurate tyre model is needed. This involves parameter estimation of a large amount of tyre parameters, when a Magic Formula is employed. An accurate model of the steering wheel torque relation to steering angle is needed as well. Here, the assistance torque provided by the power steering module needs to be taken into account as well.

Furthermore, the choice of the NLP solver is vital when increasing the problem size. The relative performance of IPM and SQP differs greatly for an increase in problem size.

8-1-3 Implementation with additional driver applied torque

MPC and human driver behaviour are very analogous to each other. Human drivers do not only take into account current time steps, but also future time steps. At the same time, human drivers may plan to carry out a manoeuvre that results in a loss of stability. Similar to an MPC problem, this can be regarded as a an infeasible manoeuvre.

The predictive nature of MPC may very well be helpful in critical situations where drivers have trouble finding an optimal solution to avoid a crash. The MPC can be used to assess the risk of different situations and take measures in time. Depending of the the situation, driver intervention or assistance can be applied, just as current Evasive Manoeuvre Assistance Function (EMA) functions do. The advantage of MPC is that the driver can be assisted at an earlier stage, which can be crucial in near-collision situations.

8-1-4 Longitudinal inputs on an experimental setup

The combined steering and braking scenario is not tested on an experimental setup in this thesis. It was found that SQP has more trouble handling a combined steering and braking scenario than IPM does. Combined steering and braking might prove to be critical in near-collision situations. Not only braking could be implemented, but also positive torque on each wheel, essentially yielding torque vectoring. This allows for quick and agile turns which could be very useful in critical situations.

8-1-5 Obstacle dependent cost function for collision avoidance

In this thesis a fixed baseline scenario was defined for a rear-collision situation. In reality, an infinite amount of near-collision scenarios could occur. For practical implementation, it would be desired that the presence of obstacles is considered in the cost function. This way, an optimal trajectory can be determined, based on the position of different obstacles.

8-1-6 Experimental tests with higher steering wheel torque limits

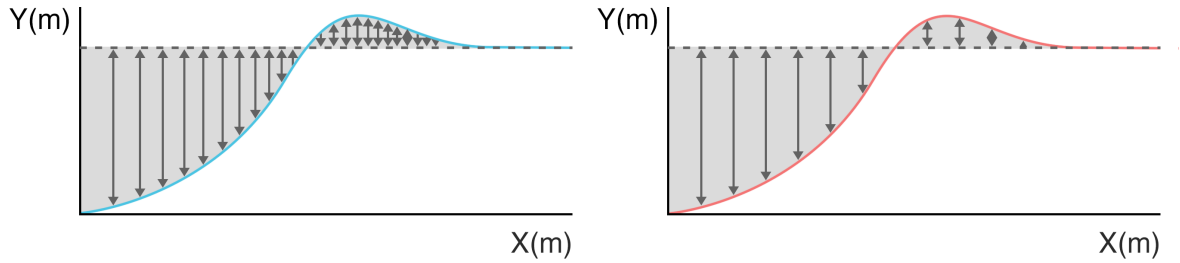
Along with the transition to autonomous driving, power steering modules in vehicles will be able to provide higher steering wheel torques. This means that the actuator limits are set higher than 7 Nm, allowing for more aggressive manoeuvres and the ability to truly utilise the nonlinear part of the tyre in limit handling scenarios.

8-1-7 Embedded implementation of IPM

For larger sized problems, IPM solved by IPOPT becomes a more attractive approach. The commercial ForcesPro packages allows for self contained C code generation that can be deployed on embedded systems, using IPM. To extend the MPC scheme of this thesis to problems that involve more advance vehicle models, a C coded IPM solver may prove to be a promising solution for an embedded system within a vehicle.

8-1-8 Spatial formulation for MPC

It was found that for the combined braking and steering scenario, brakes were only used if the longitudinal velocity v_x was penalised in the cost function. The goal was to move towards a reference position as fast as possible, while decreasing v_x as much as possible. Naturally, this is a conflicting objective. Due to the fact that the optimisation is run over time, the MPC is incentivised to be at the reference position as quickly as possible. Otherwise, the cost will build up, as shown in figure 8-2a. It can be seen that more samples are obtained as the vehicle slows down. The cost sums the distance over all simples and will therefore increase once there are more samples. Thus, no braking is applied without a v_x term in the cost function. If the dynamics are dependent on the position of the vehicle, slowing down is not penalised anymore (figure 8-2b). The intervals whose sum define the cost function have an equal distance to each other. Examples of spatial MPC formulations can be found in [18] and [42].



(a) Time dependent formulation

(b) Spatial formulation

Figure 8-2: Influence of formulation type on cost for combined steering & braking

Appendix A

Simulation results

A-1 SQP & IPM (steering only) compared at computational limits

In sections 6-1 and 6-2 the steering only scenario was executed by means of both Sequential Quadratic Programming (SQP) and the Interior-Point Method (IPM). It was found that SQP resulted in significantly shorter computation times. To directly compare two methods that take around 0.15 s (IPM) and $1.5 \cdot 10^{-3}$s (SQP) would be unfair. Therefore, in this comparison both methods are tuned such that they run at the limits of what the computation time allows for.

A-1-1 SQP & IPM (steering only) formulation

The cost functions used in this chapter are listed below. The cost function for IPM remains the same cost function used for the steering only scenario in section 6-1.

$$J_{IPM} = \sum_{k=1}^{N} 2(Y(t+k) - Y_r(t+k))^2 + u(t+k)^2 \tag{A-1}$$

Due to the fact that the Model Predictive Control (MPC) time parameters (table A-1) for SQP are considerably different from those used in section 6-2, a different cost function is used for SQP:

$$J_{SQP} = 10^6(Y(N) - Y_r(N))^2 + \sum_{k=1}^{N} 5 \cdot 10^{-6}(Y(t+k) - Y_r(t+k))^2 \tag{A-2}$$

The MPC parameters for IPM and SQP found in table A-1 were found to match each other in terms of performance. The computation times for both these tunings in simulation were found to be just below the sampling time to ensure that both methods are close the limits of the processor's real-time capabilities. The sampling time for IPM is decreased to 0.1s to

Table A-1: MPC parameters

	IPM	SQP
N	20	160
T_s	0.1 s	0.02 s
H	2 s	3.2 s
h	0.1 s	0.02 s

balance out the advantage SQP has as it runs directly in C code. In section A-2 computation times were found to be around 0.15 s. It is expected that IPM will speed up sufficiently to run at 0.1 s if it runs directly in C code. Computation times for generated code in CasADi are generally 4 to 10 times shorter [29]. As in all previous simulation runs, the initial velocity is set to $v_{x_0} = 20\,\mathrm{m/s}$.

A-1-2 Trajectory

The resulting trajectories for both methods are shown below:

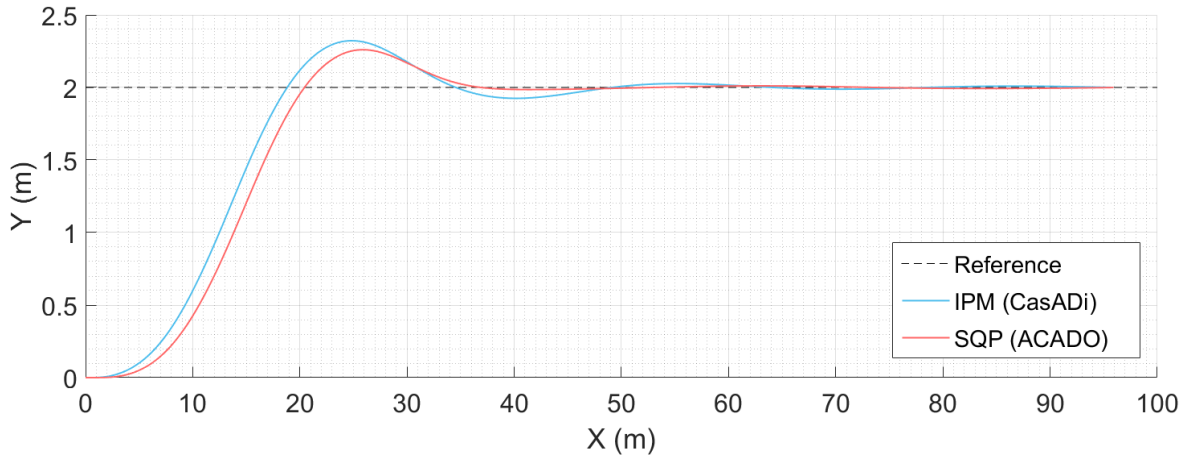


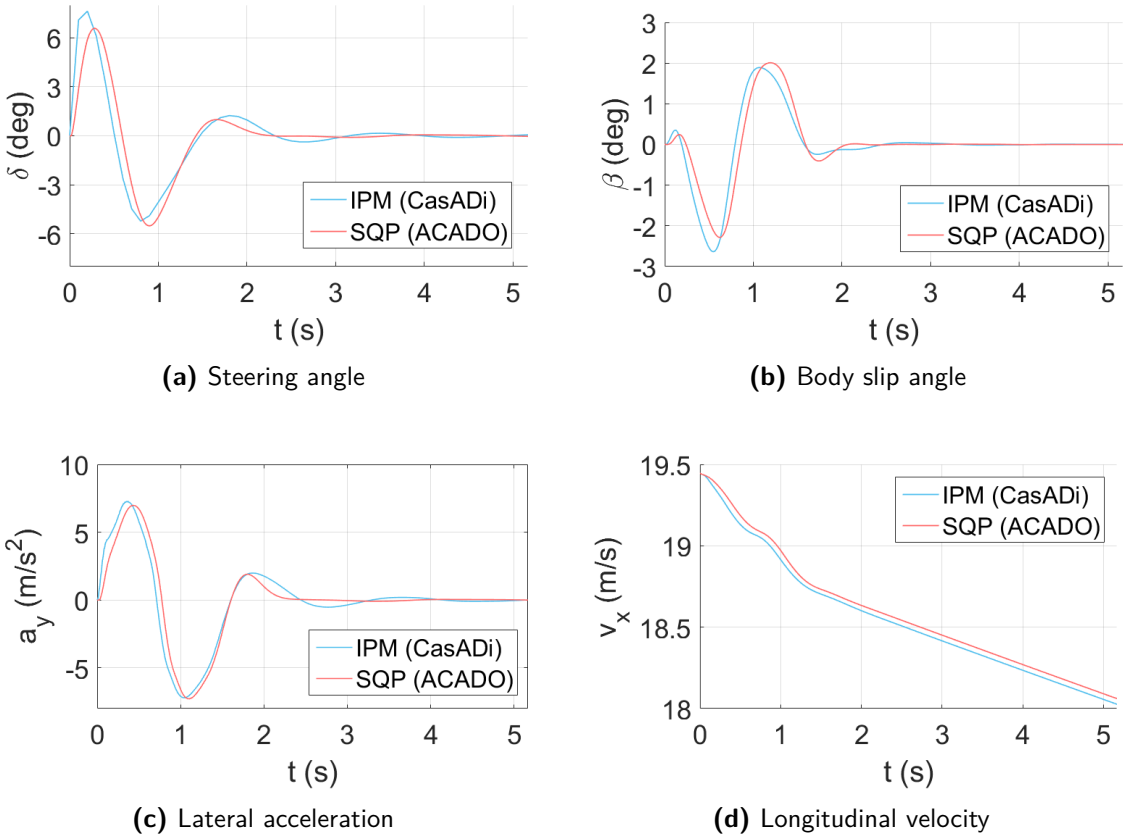
Figure A-1: Comparison of trajectories of steering only scenario for SQP and IPM

The undershoot that is present in the IPM method can be attributed to the absence of a terminal cost term in CasADi. SQP on the other hand, penalizes the terminal states much more than the states over the horizon. Therefore the MPC is given incentive to try to stay close to the reference position at the end of the horizon, resulting in no undershoot.

A-1-3 Notable results

The remaining measurements are shown in figure A-2. It can be seen that SQP and IPM show nearly identical behaviour for these tunings. The identical behaviour helps to validate that both methods do indeed yield an optimal trajectory for the posed problem.

Compared to the previous tuning for IPM in section A-2, a much smoother response is obtained by decreasing the sampling time to 0.1s. It does make for a less aggressive manoeuvre with a considerably lower steering angle and body slip.



(a) Steering angle

(b) Body slip angle

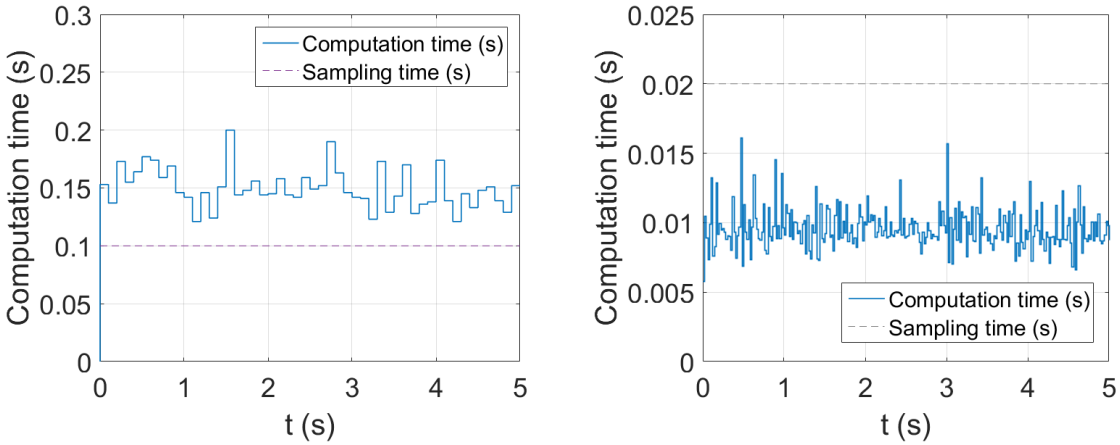
(c) Lateral acceleration

(d) Longitudinal velocity

Figure A-2: Comparison of results for SQP and IPM

A-1-4 Computation times

The computation times for both methods are shown below. The mean computation times $t_{c,mean}$ for IPM and SQP are 0.147 s and 0.0096 s, respectively.



(a) Computation times (IPM)

(b) Computation times (SQP)

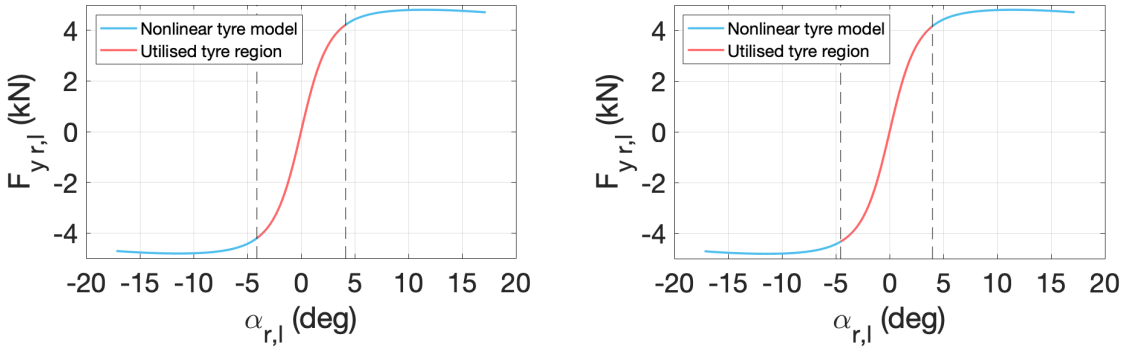
Figure A-3: Computation times for SQP and IPM

A-1-5 Friction utilisation

The reason for employing Nonlinear Model Predictive Control (NMPC) was to be able to utilise the nonlinear region of the tyre. In this section it is assessed whether the nonlinear part of the tyre is used indeed. The tyre model within the MPC is a simplified Magic Formula, as shown in section 3. In the vehicle model that CarMaker uses, a full 5.2 Magic Formula [43] is used. As there is a mismatch between the model within the MPC and the highly realistic CarMaker vehicle model, both models yield different slip angles and tyre forces. In this section, the utilised friction for both the model within the MPC as well as simulations in CarMaker is considered.

In figure A-3 the measured slip angles are shown for the rear left wheel in CarMaker. They are plotted against the lateral tyre forces, using the simplified Magic Formula used in the MPC, as shown in equation (3-10).

This figure displays to what extent the nonlinear part of the tyre is being utilised. For both SQP and IPM the nonlinear part of tyre is clearly utilised. The maximum slip angles for both methods are still somewhat off the tyre saturation point. However, it can not be assumed that larger slip angles will result in better performance. The MPC runs an optimisation over time. By increasing the slip on the wheels, more longitudinal velocity is lost while executing the manoeuvre. This results in the car taking more time to reach to its reference position compared to a situation with less slip. Therefore, the MPC is more likely to opt for a manoeuvre with slightly lower slip angles.



(a) SQP **(b)** IPM

Figure A-4: Friction utilisation of rear left wheel for SQP and IPM

The utilised friction for both methods is very similar. The maximum slip angle for IPM is slightly higher than for SQP. SQP is more present at the lower slip angles whereas IPM utilises the higher slip angles slightly more. The slip angle frequency distribution for both methods is shown in figure A-5. It can be seen that IPM utilises large slip angles slightly more than SQP.

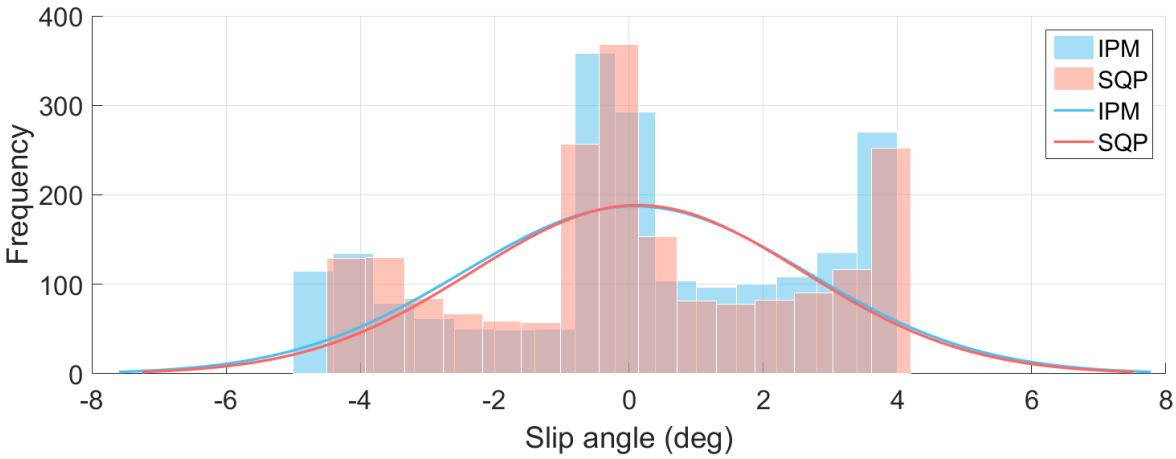
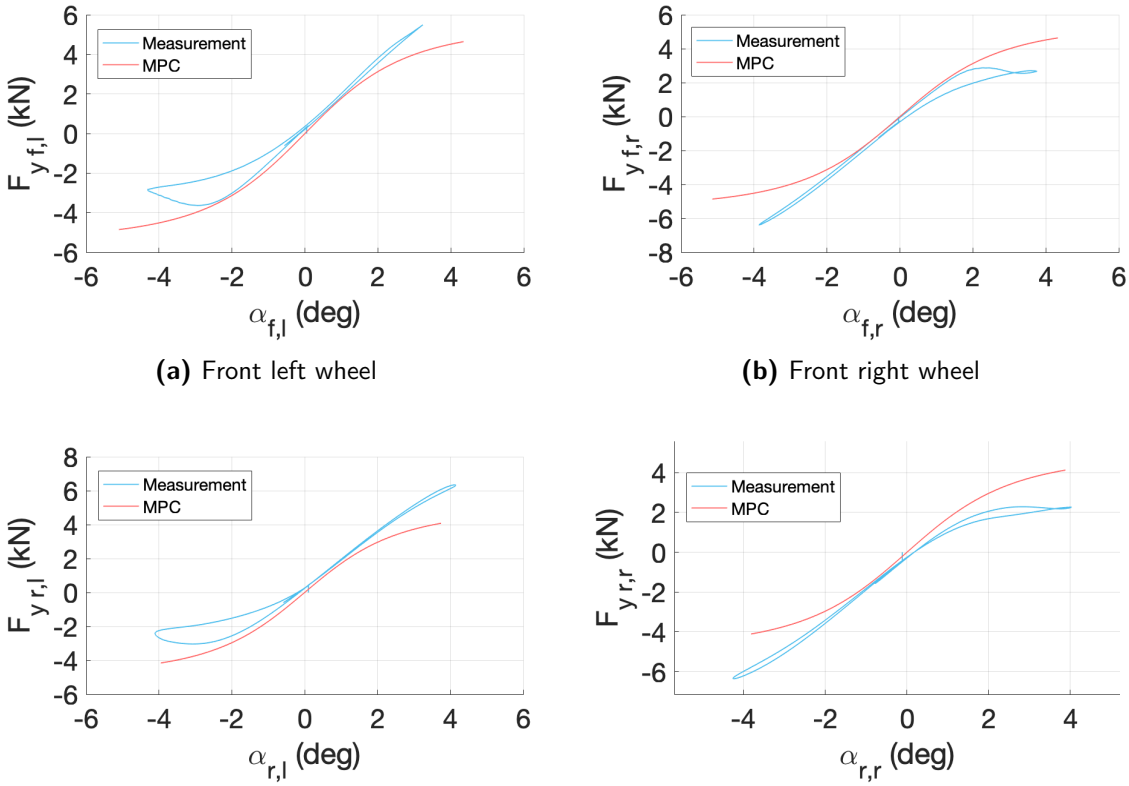


Figure A-5: Obtained slip angles for SQP and IPM

In figure A-6 the slip angles with corresponding lateral tyre forces for the MPC are shown. They are compared with the slip angles and lateral tyre forces that are both measured in CarMaker. It is clear that the MPC does not consider load transfer, resulting in changing normal forces on each wheel. Load transfer has a relatively large impact on the left two wheels, especially. The lateral tyre forces is directly proportional to the normal force on the tyre. For the negative slip angles for the left two wheels it can be seen that the varying normal force has a significant impact on the generated lateral tyre force. For the same slip angle, different lateral tyre forces are generated.



(a) Front left wheel

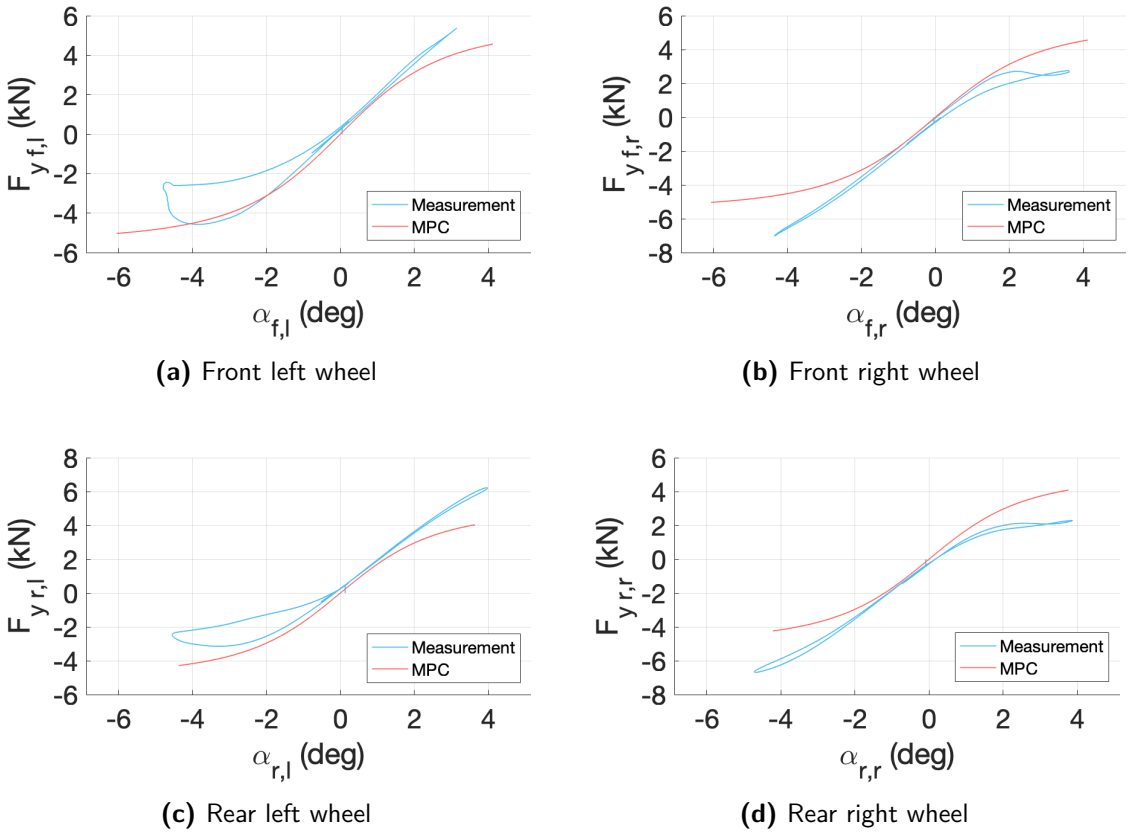
(b) Front right wheel

(c) Rear left wheel

(d) Rear right wheel

Figure A-6: Lateral tyre forces and slip angles for SQP

Similar behaviour can be observed using IPM, as shown in figure A-7.



(a) Front left wheel

(b) Front right wheel

(c) Rear left wheel

(d) Rear right wheel

Figure A-7: Lateral tyre forces and slip angles for IPM

A-2 IPM (steering only)

In this section, the IPM method shown in section 6-1 is shown individually for the steering only scenario.

A-2-1 Trajectory

The trajectory for the steering only scenario using IPM is shown in figure A-8.

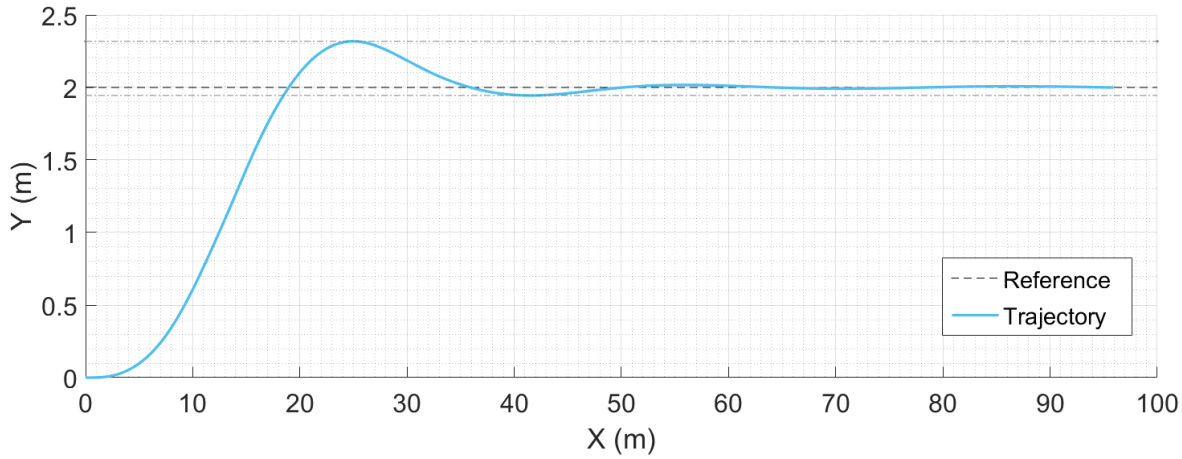
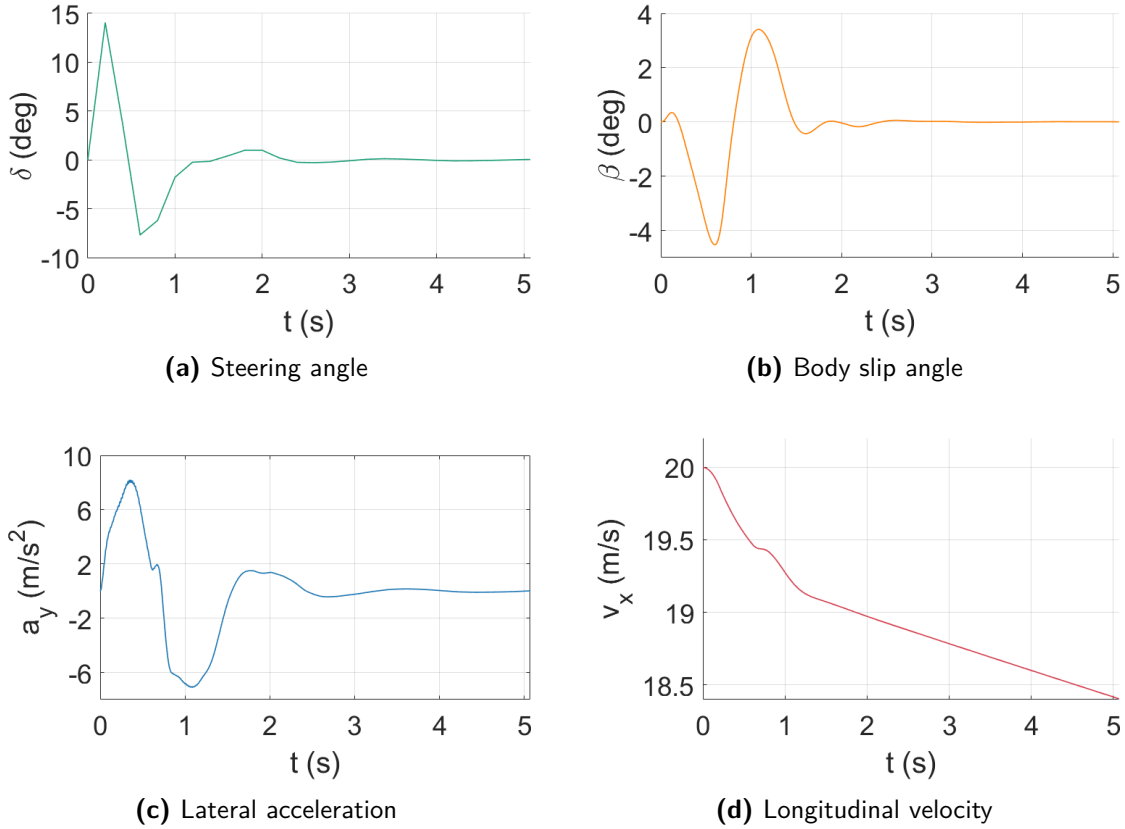


Figure A-8: Trajectory of steering only scenario using IPM

A-2-2 Notable results

The most notable results for the steering only (IPM) scenario are shown here:



(a) Steering angle

(b) Body slip angle

(c) Lateral acceleration

(d) Longitudinal velocity

Figure A-9: Results of steering only scenario using IPM

A-2-3 Computation times

The computation times of all iterations are shown in figure A-10. It can be seen that the computation times remain below the sampling time. This ensures that it is possible to run the MPC in real-time. The time listed here is the total time it takes to setup the Nonlinear Programme (NLP) problem and time spent in the NLP solver, at each iteration. The maximum computation time can also be constrained to a maximum allowable amount. This may however lead to solutions that are not optimal. In this case, the computation time is unconstrained.

A-2-4 Friction utilisation

The utilised friction can be observed in figure A-11.

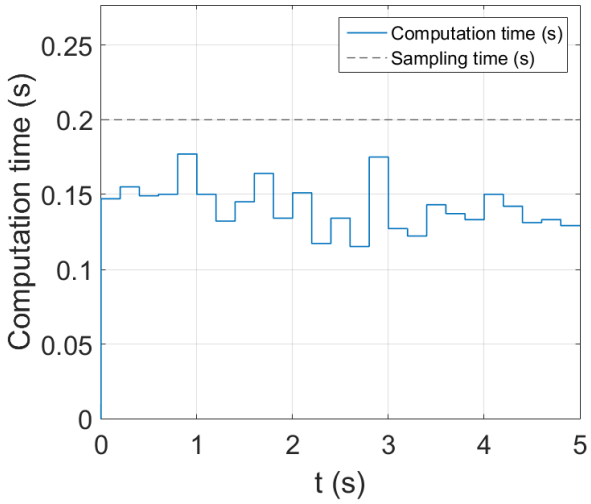
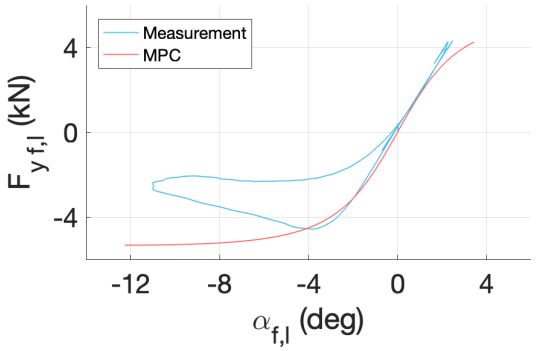
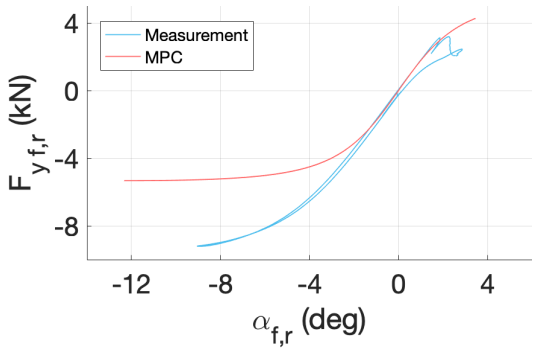


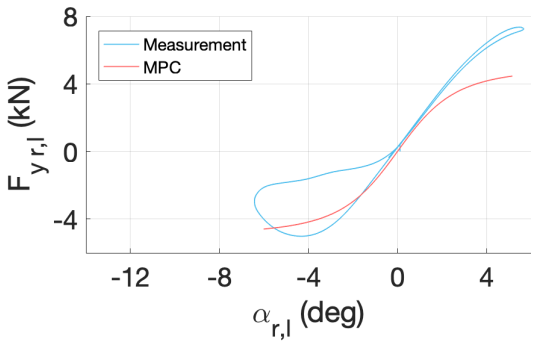
Figure A-10: Computation time of steering only scenario using IPM



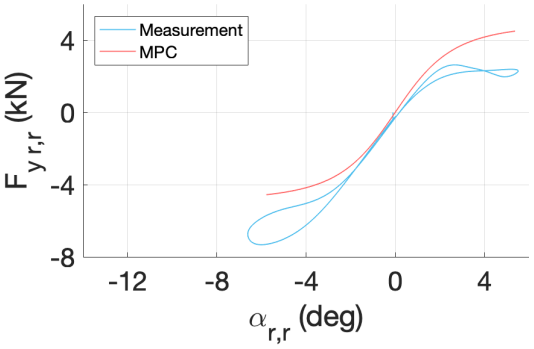
(a) Front left wheel



(b) Front right wheel



(c) Rear left wheel

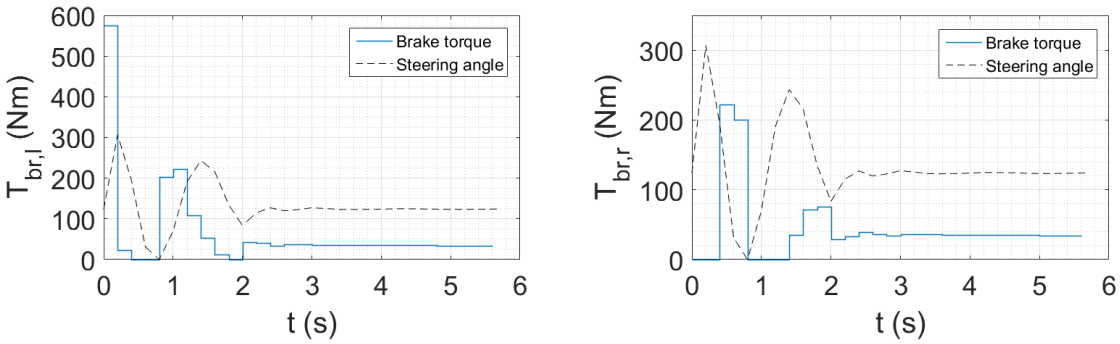


(d) Rear right wheel

Figure A-11: Utilised friction for steering only scenario using IPM

A-3 IPM (Steering & braking)

The front brake torques were shown in figure 6-5. The rear brake torques show a similar behaviour and can be observed in figure A-12.



(a) Rear left wheel **(b)** Rear right wheel

Figure A-12: Brake torques at the rear wheels

A-3-1 Computation times

The computation times are shown in figure A-13. Due to the increase in problem size compared to the steering only scenario, computation times have increased. The computation times reach just above the sampling time for some iterations. Therefore, the execution time for posing and solving the NLP needs to be limited to run in real-time. This will decrease the performance and is not desired for complex scenarios where steering and braking are combined.

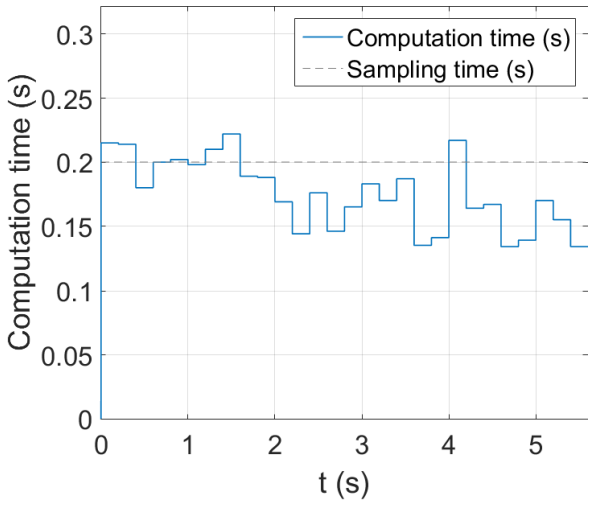
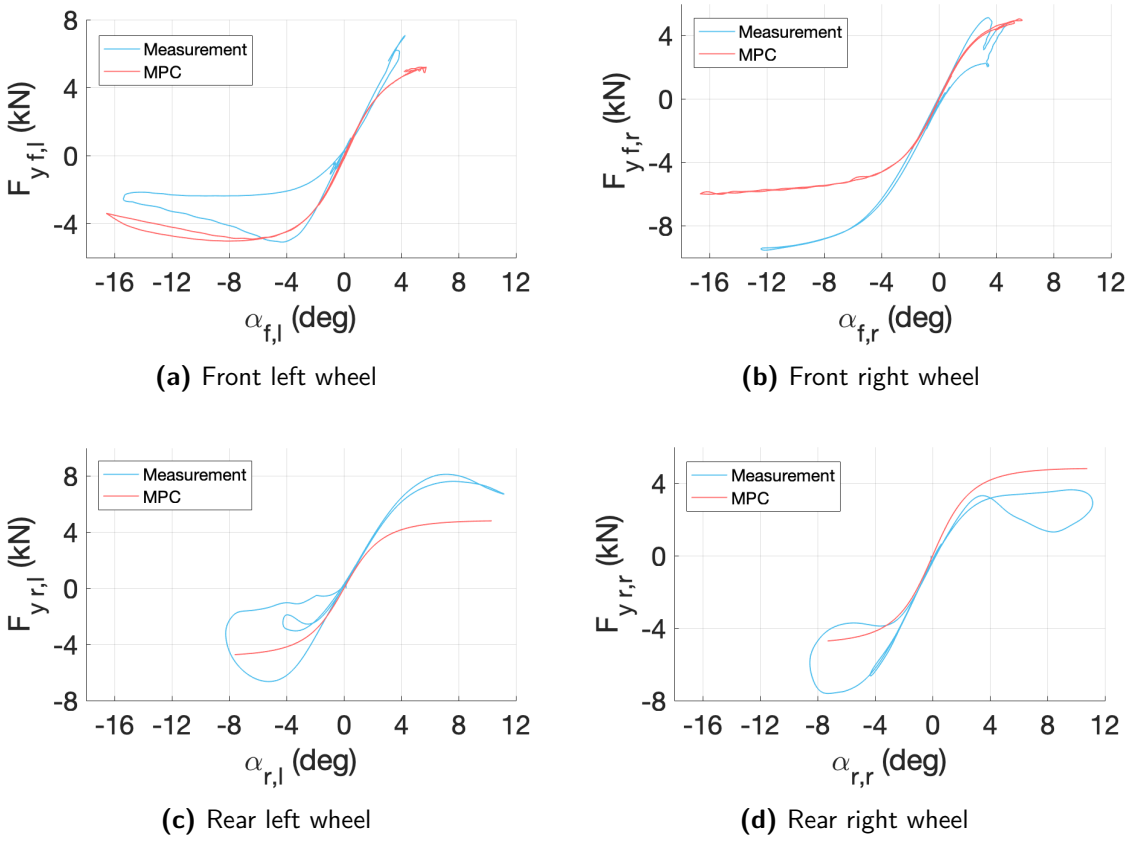


Figure A-13: Computation time of steering & braking scenario using IPM

A-3-2 Friction utilisation

The utilised friction for the steering and braking scenario (IPM) can be seen in figure A-14. The simplified Magic Formula does not incorporate any combined slip. Longitudinal and lateral slip influence each other, so a solely lateral tyre model lacks accuracy for a combined slip scenario. More advanced tyre models incorporate combined slip behaviour. Along with the previously explained lack of load transfer, a significant model mismatch is present here.



(a) Front left wheel

(b) Front right wheel

(c) Rear left wheel

(d) Rear right wheel

Figure A-14: Utilised friction for steering & braking scenario using IPM

A-4 SQP (steering only)

A-4-1 Computation times

The computation times for SQP are shown in figure A-15. Clearly, computation times are well below the sampling time (0.04s). ACADO and SQP are known to perform efficiently for smaller sized problems, such as is the case in this scenario.

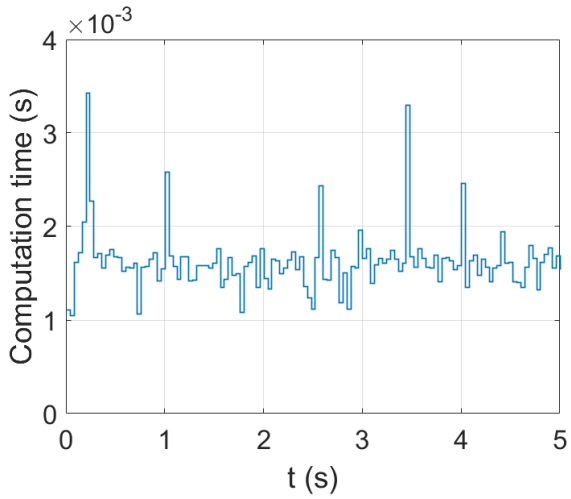
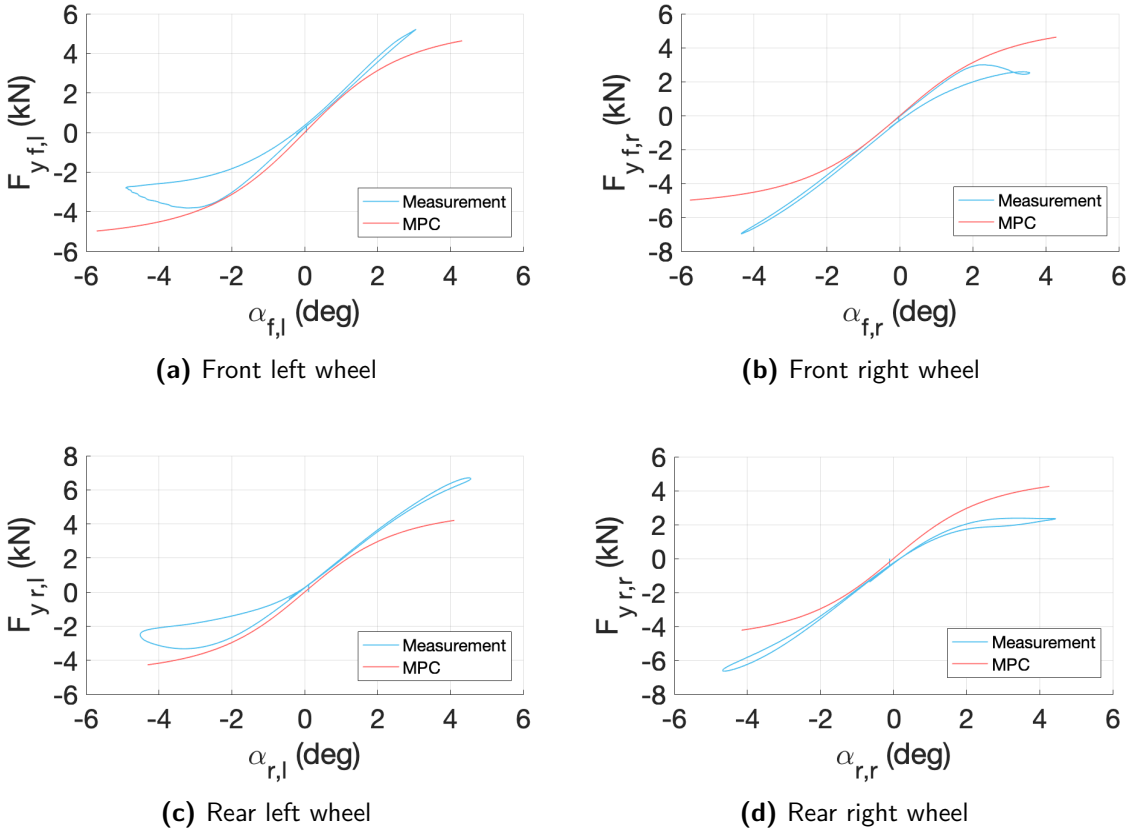


Figure A-15: Computation time of steering scenario using SQP

A-4-2 Friction utilisation

The friction utilisation is shown in figure A-16.



(a) Front left wheel

(b) Front right wheel

(c) Rear left wheel

(d) Rear right wheel

Figure A-16: Utilised friction for steering only scenario using SQP

A-5 Scalability to different scenarios

In this section a comparison is made between the use of different reference positions in the cost function and the influence of different initial velocities on the performance of the MPC. The aim here is to see how well the MPC scales when a slightly different problem is posed. It must be noted that for each objective the MPC can be retuned. However, in real-life driving scenarios there are are an infinite amount of possible scenarios and it is cumbersome to tune for each different scenario. Therefore, in this section the MPC formulation remains the same throughout the different scenarios so that it can be observed how well the MPC scales with different problems. Of course, scalability may differ among different cost functions. An aggressively tuned cost function is more likely to fail when it is applied to a different problem. The results in this section are aimed to reflect the general scalability that was observed for different tunings of both SQP and IPM during this thesis. The cost functions remain the same as in (6-2) and (6-5).

In the reference comparisons only the global reference position Y_r is changed. The weights in the cost function remain the same. In the initial velocity comparison the cost function remains completely the same and only the initial velocity is adjusted.

A-5-1 Comparison of multiple references

In this section different reference positions are compared for both NLP methods.

SQP

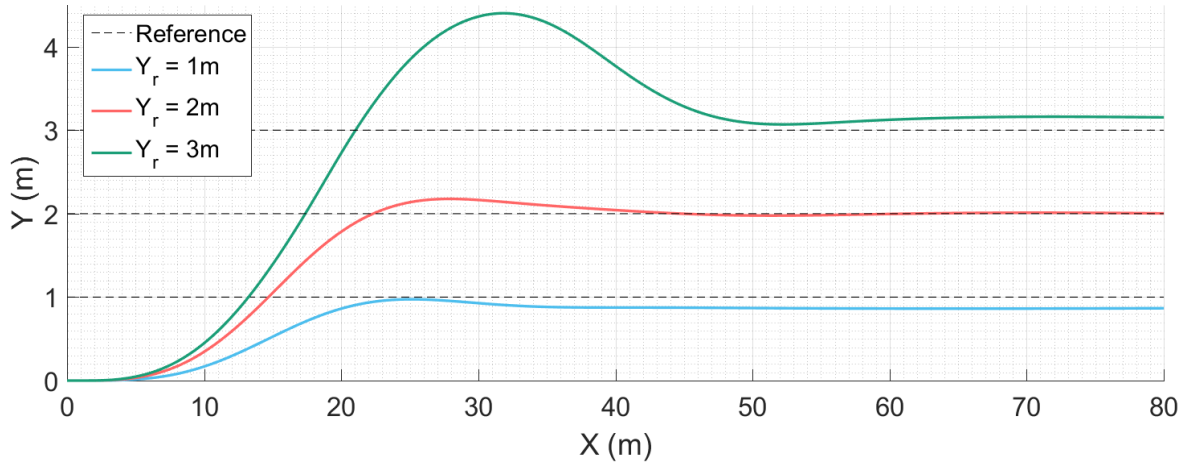


Figure A-17: Scalability of reference position using SQP ($N = 40$)

In figure A-17 the same MPC is used as as in section 6-2-1. It can be seen that the MPC does not scale well for different reference positions as it fails to track $Y_r = 1m$ and $Y_r = 3m$ with the same performance as $Y_r = 2m$, which it was tuned for. There is a large overshoot present for $Y_r = 3m$ and an offset for $Y_r = 1m$. However when the horizon is increased to $N = 80$, SQP scales much better for different reference positions, as can be seen in A-18.

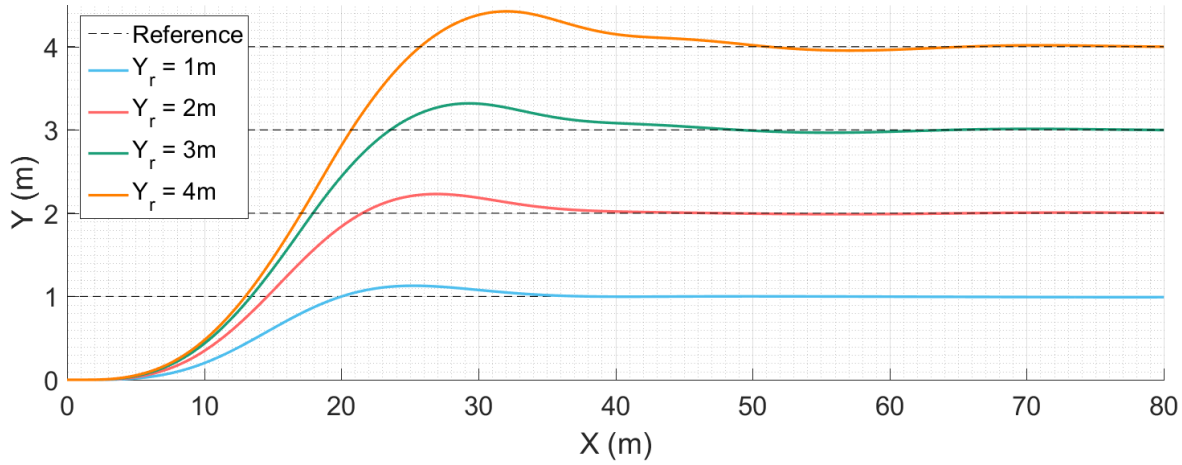


Figure A-18: Scalability of reference position using SQP ($N = 80$)

IPM

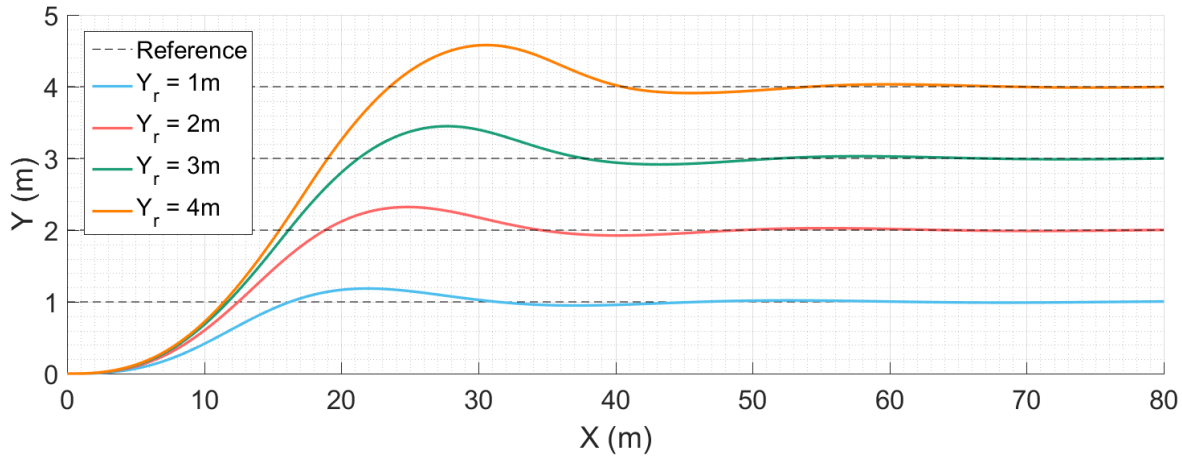


Figure A-19: Scalability of reference position using IPM

IPM, on the other hand, scales very well with different reference positions for the baseline tuning found in section A-2. The overshoot scales proportional to the given reference and it is possible to track $Y_r = 4\,\mathrm{m}$ with ease.

A-5-2 Comparison of multiple initial velocities

In order to show the versatility of the proposed MPC scheme, the trajectories for different initial velocities are shown in this section.

IPM

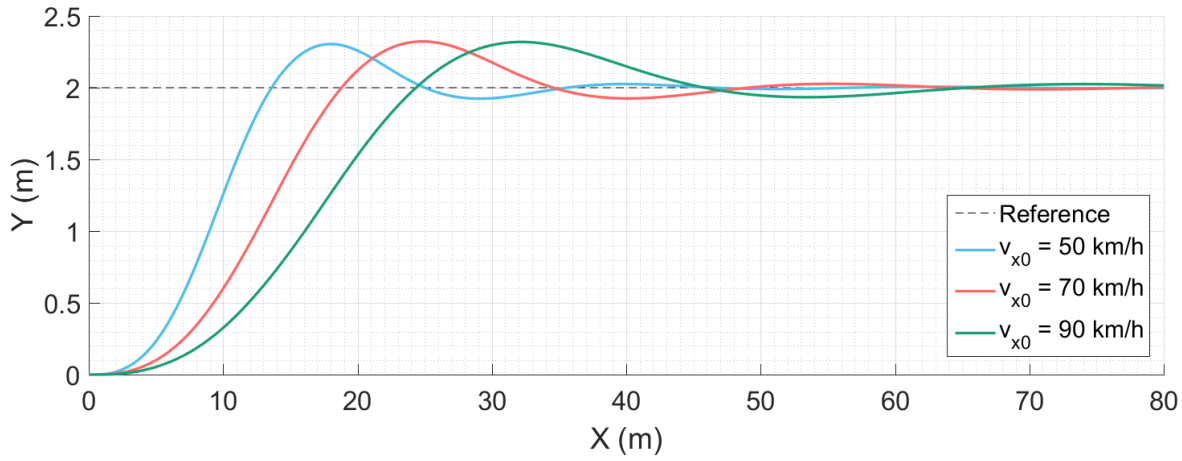


Figure A-20: Scalability of initial velocity using IPM

IPM is able to scale well with different inital velocities as well. As seen before, IPM reaches the 2m safe-zone slightly faster than SQP.

SQP

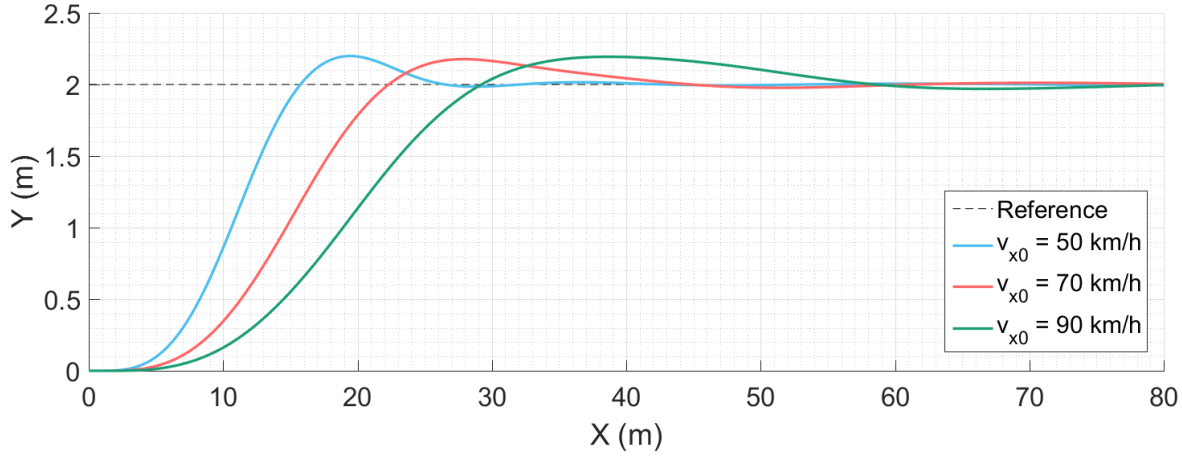


Figure A-21: Scalability of initial velocity using SQP

SQP is able to track the reference at different initial velocities without the need for increasing the amount of control intervals.

A-6 Horizon comparison

The time horizons considered in the previous sections were kept constant at 2 s and 1.6 s for IPM and SQP, respectively. In this section, both time horizons H are increased twofold.

A-6-1 IPM

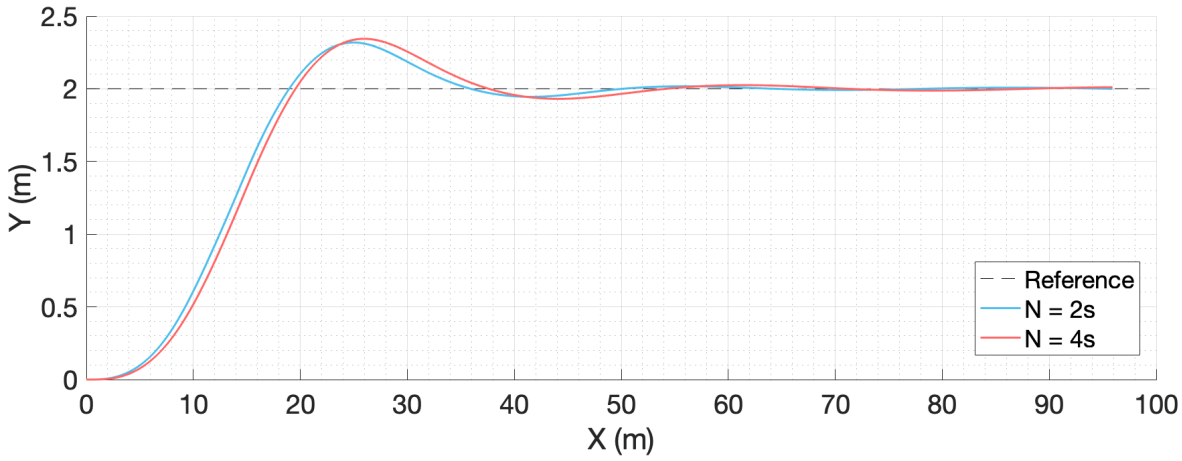


Figure A-22: Comparison of different horizons using IPM

It can be seen that an increase in the horizon makes for a slightly less aggressive manoeuvre. This can be attributed to the fact that for a short horizon only the first part of the manoeuvre is considered. For a longer horizon, the weight of the first part of the manoeuvre is relatively lower, resulting in a more relaxed manoeuvre.

A-6-2 SQP

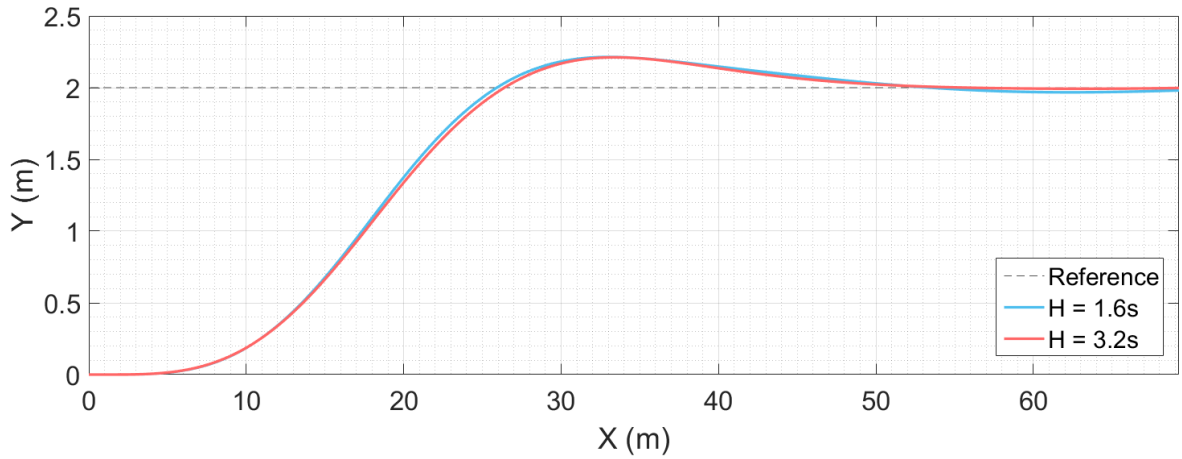


Figure A-23: Comparison of different horizons using SQP

For SQP the difference between the two time horizons is even smaller than for IPM. Again, it can be seen that a shorter time horizon results in a slightly more aggressive manoeuvre.

A-7 Sampling time comparison

In this section, different sampling times are compared for both methods.

A-7-1 IPM

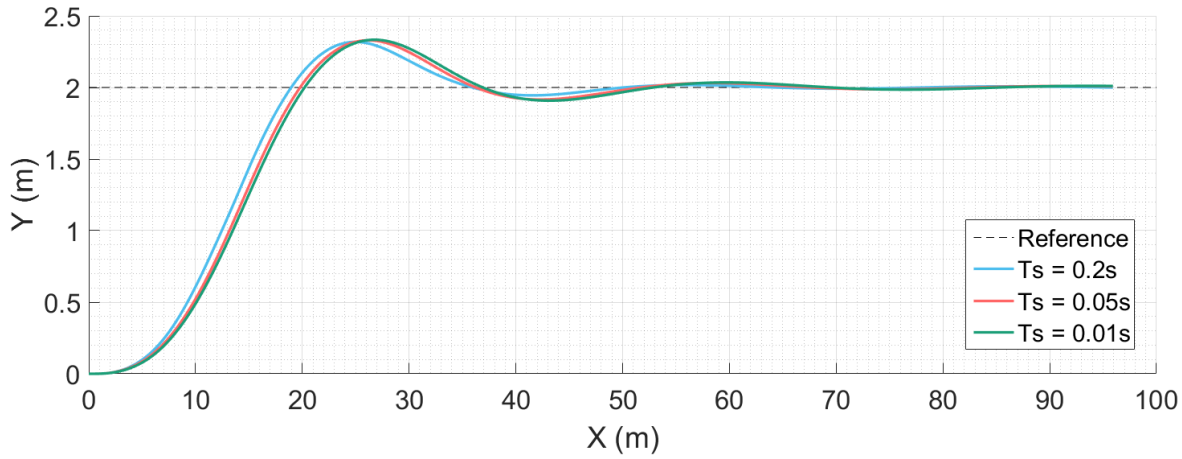


Figure A-24: Comparison of different sampling times using IPM

Decreasing the sampling time does not have much influence for IPM. Shorter sampling times result in slightly less aggressive manoeuvres. It must be noted that the time horizon here is kept constant for different sampling times.

A-7-2 SQP

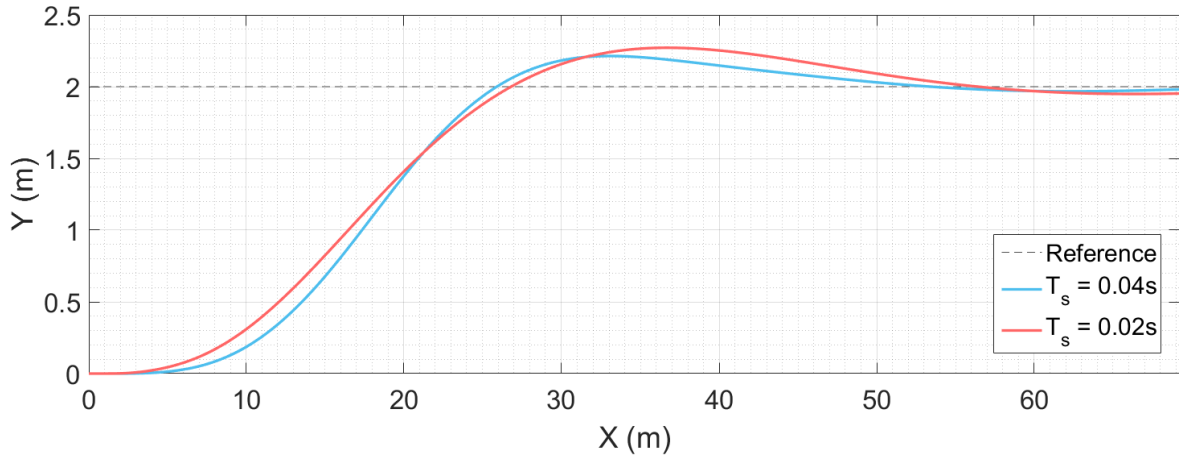


Figure A-25: Comparison of different sampling times using SQP

It must be noted that decreasing the sampling time to 0.01 s did not result in convergence to the reference position. This can be attributed to the fact that the amount of control intervals remains unchanged for the different sampling times shown here, resulting in a very short time horizon. When $N = 40$, a sampling time of 0.04 s results in $H = 1.6$ s, whereas $T_s = 0.01$ s results in $H = 0.4$ s. If the time horizon is kept constant as was done using IPM, it is likely that behaviour for the different sampling times is similar as for IPM. The MPC is not able to converge to the reference for such short time horizons. It can be concluded that a balanced trade-off needs to be made in regard to T_s and N.

Appendix B

Experimental results

B-1 Results at 50 KPH

K_s	b_s	J_s
2.5	0.9	0.05

Table B-1: Torque model parameters at $v_{x0} = 50\text{km/h}$

$a_{y,max}$	$a_{y,min}$	β_{max}	β_{min}	Y_{max}	Y_{min}	X_s
$4.08\,\text{m/s}^2$	$-5.53\,\text{m/s}^2$	$1.17\,\text{deg}$	$-1.59\,\text{deg}$	$2.125\,\text{m}$	$1.779\,\text{m}$	$21.000\,\text{m}$

Table B-2: Experimental results at $v_{x0} = 50\text{km/h}$

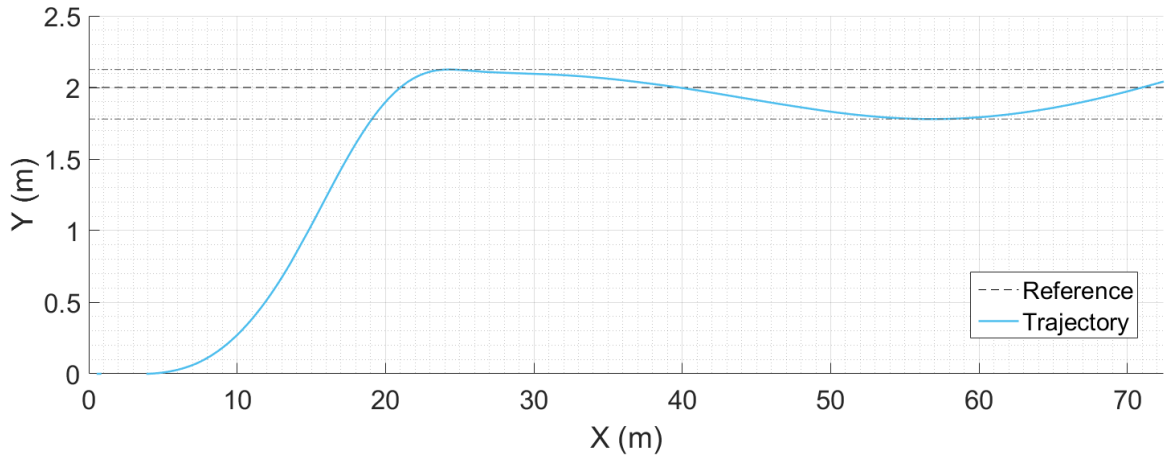
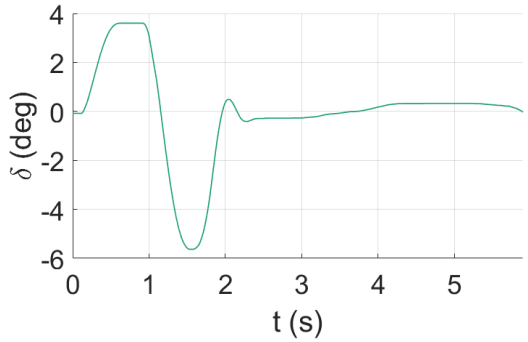
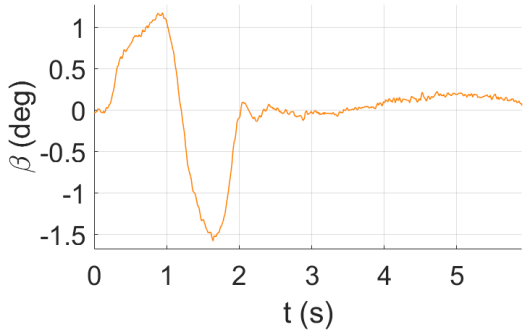


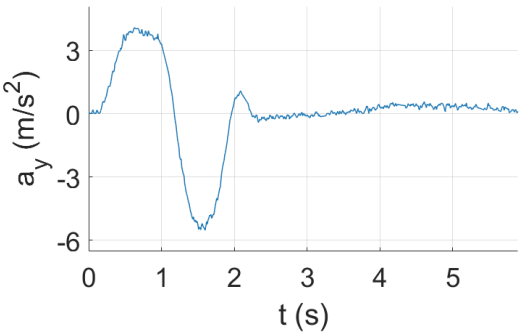
Figure B-1: Trajectory of experimental test at $v_{x0} = 50\text{km/h}$



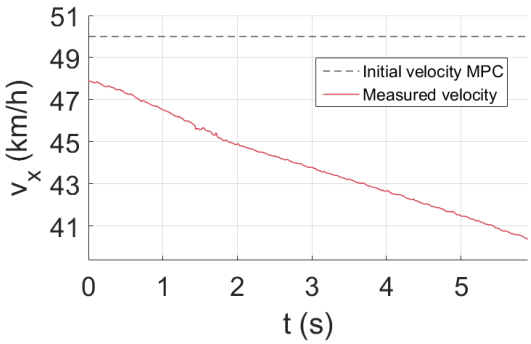
(a) Steering angle



(b) Body slip angle



(c) Lateral acceleration



(d) Longitudinal velocity

Figure B-2: Results of experimental test at $v_{x0} = 50\text{km/h}$

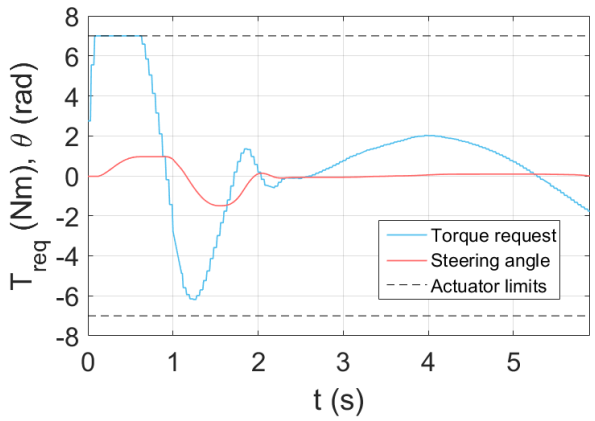


Figure B-3: Requested torque and steering wheel angle at $v_{x0} = 50\text{km/h}$

B-2 Results at 60 KPH

K_s	b_s	J_s
3.5	0.9	0.05

Table B-3: Torque model parameters at $v_{x0} = 60\text{km/h}$

v_{x0}	$a_{y,max}$	$a_{y,min}$	β_{max}	β_{min}	Y_{max}	Y_{min}	X_s
$5.03\,\text{m/s}^2$	$-6.45\,\text{m/s}^2$	$0.80\,\text{deg}$	$-0.99\,\text{deg}$	$2.267\,\text{m}$	$1.862\,\text{m}$	$23.498\,\text{m}$	

Table B-4: Experimental results at $v_{x0} = 60\text{km/h}$

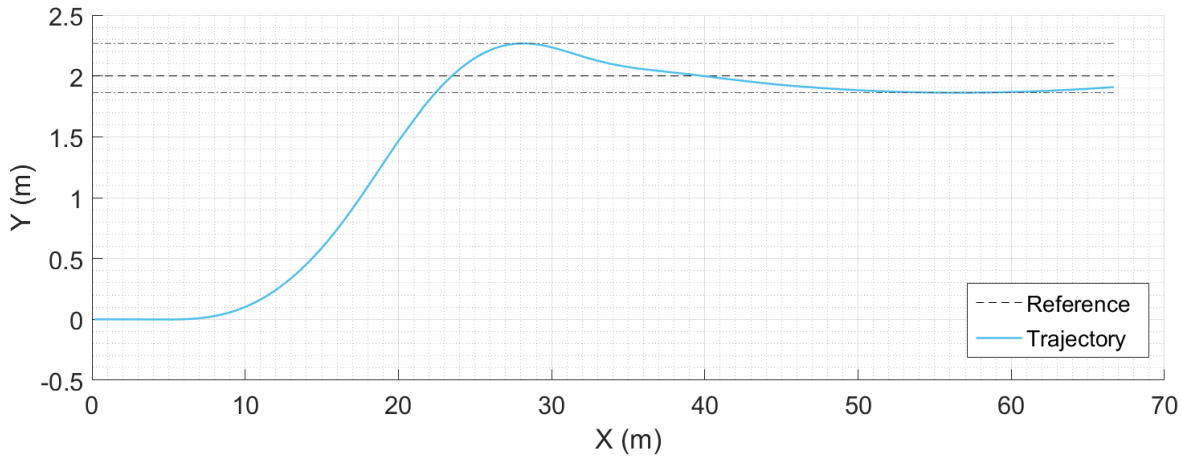
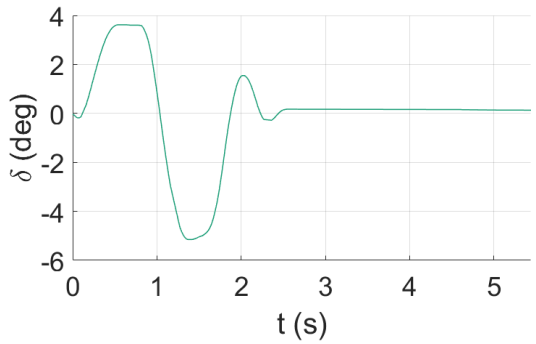
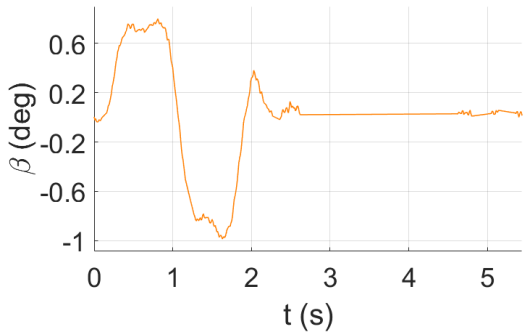


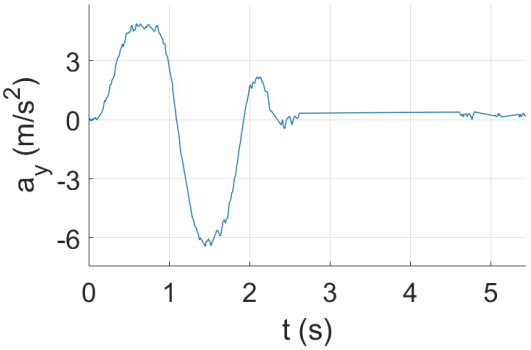
Figure B-4: Trajectory of experimental test at $v_{x0} = 60\mathrm{km/h}$



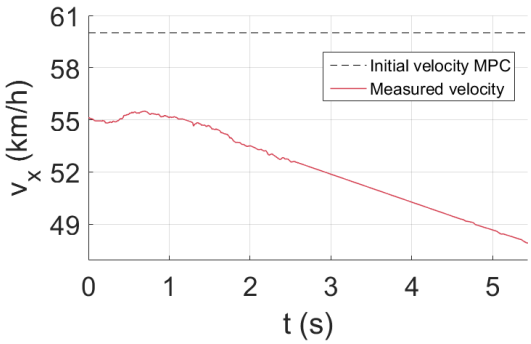
(a) Steering angle



(b) Body slip angle



(c) Lateral acceleration



(d) Longitudinal velocity

Figure B-5: Results of experimental test at $v_{x0} = 60\mathrm{km/h}$

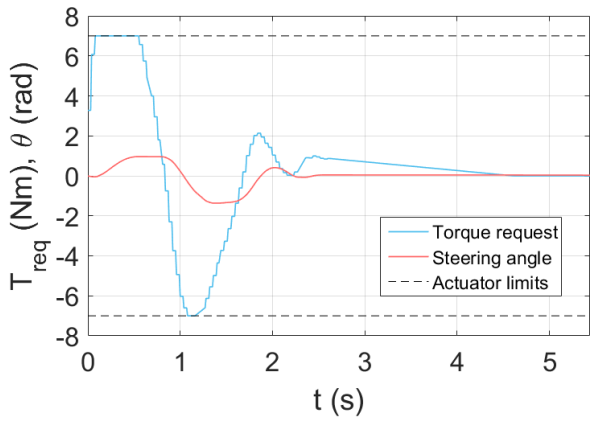


Figure B-6: Requested torque and steering wheel angle at $v_{x0} = 60\text{km/h}$

B-3 Results at 90 KPH

K_s	b_s	J_s
4.5	0.9	0.05

Table B-5: Torque model parameters at $v_{x0} = 90\text{km/h}$

$a_{y,max}$	$a_{y,min}$	β_{max}	β_{min}	Y_{max}	Y_{min}	X_s
$4.79\,\text{m/s}^2$	$-5.20\,\text{m/s}^2$	$0.40\,\text{deg}$	$-0.60\,\text{deg}$	$2.39\,\text{m}$	–	$37.283\,\text{m}$

Table B-6: Experimental results at $v_{x0} = 90\text{km/h}$

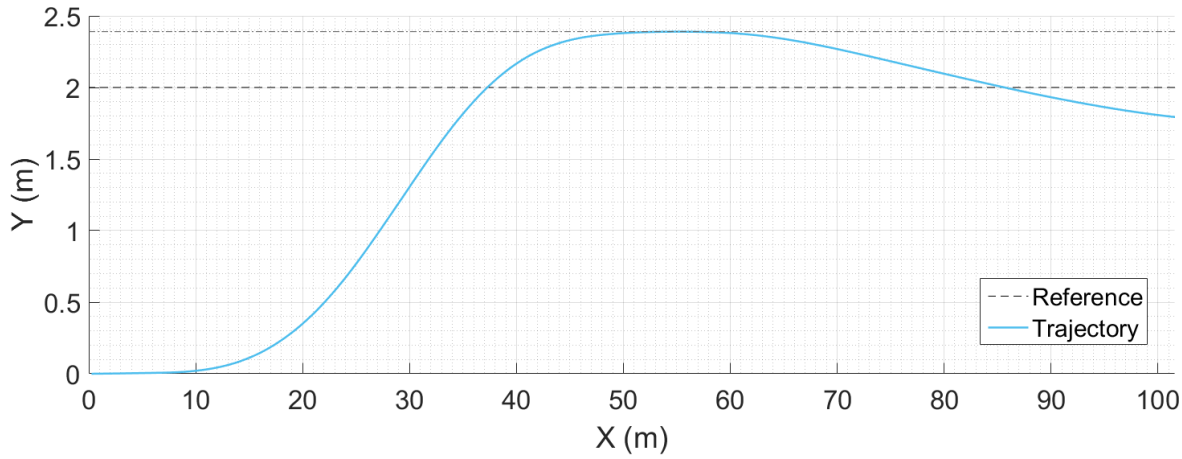
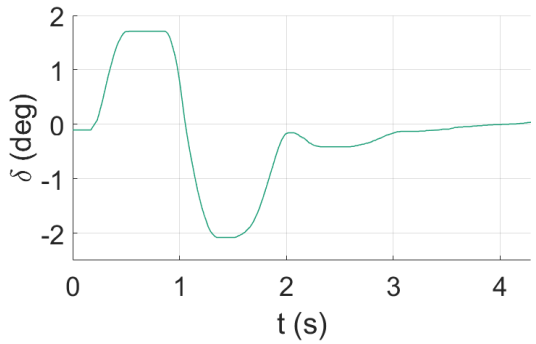
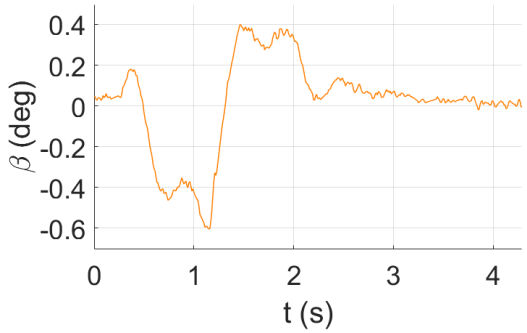


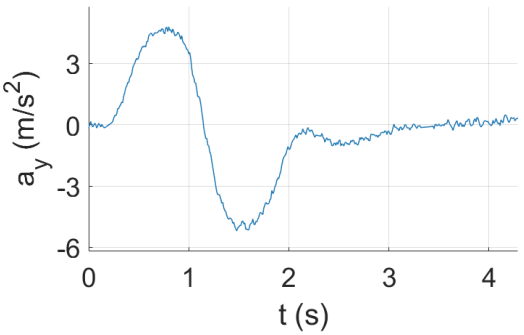
Figure B-7: Trajectory of experimental test at $v_{x0} = 90\mathrm{km/h}$



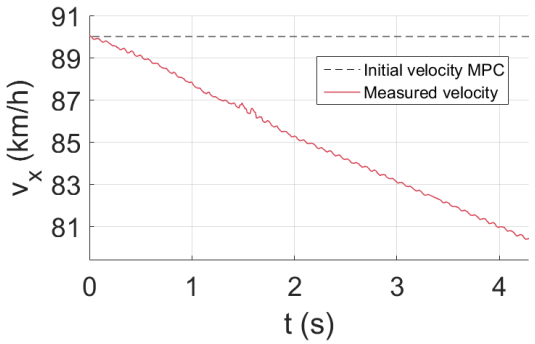
(a) Steering angle



(b) Body slip angle



(c) Lateral acceleration



(d) Longitudinal velocity

Figure B-8: Results of experimental test at $v_{x0} = 90\mathrm{km/h}$

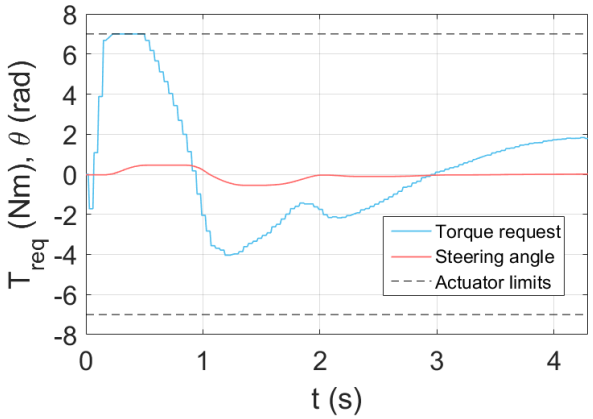


Figure B-9: Requested torque and steering wheel angle at $v_{x0} = 90\mathrm{km/h}$

B-4 Comparison between driver, simulation and experiment

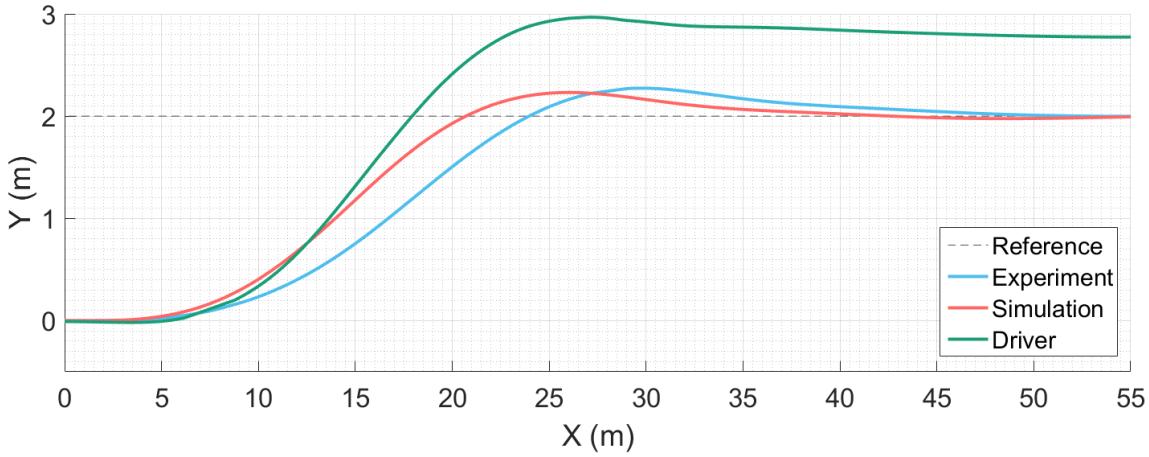


Figure B-10: Trajectory comparison for driver, simulation and experiment

While carrying out experimental tests, driver data was logged as well. A comparison of driver, simulation and experimental data can be seen in figures B-10 and B-11. No cones or any other positioning marks were used when the driver carried out the manoeuvre. The driver had no reference as to what was the actual lateral displacement. Therefore, the lateral displacement is slightly larger and it is hard to draw a fair comparison. It can be seen that a driver is able to initiate a more aggressive manoeuvre. The lateral displacement is larger for the same distance travelled. For a higher torque actuator limit, the Model Predictive Control (MPC) initiated manoeuvre would come closer to the aggressiveness of the driver.

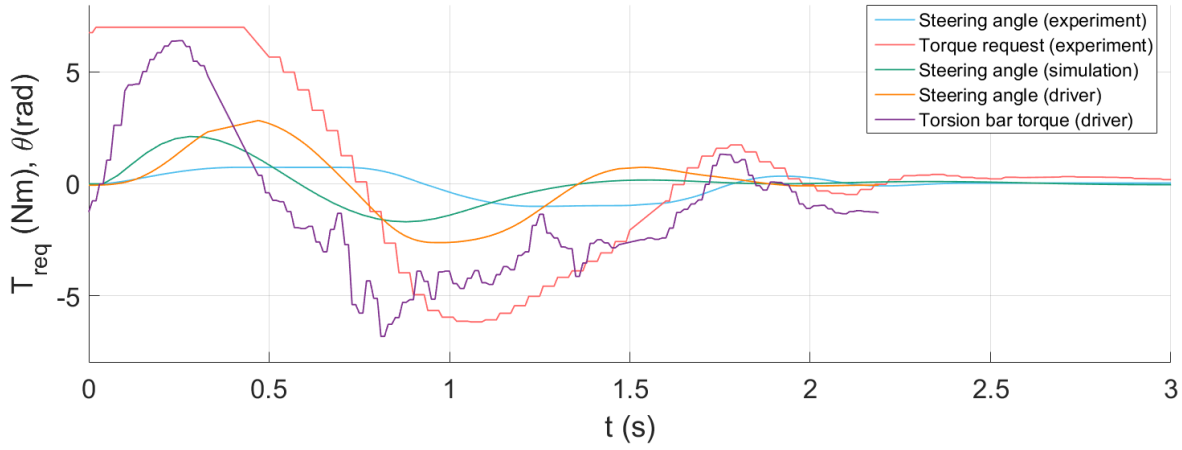


Figure B-11: Torque request and steering angle comparison for driver, simulation and experiment

In figure B-11, the steering wheel torque (experiment), torsion bar torque (driver) and steering angles are shown. For driver applied torque, only the torsion bar torque can be measured. The torque applied at the steering wheel is different from the torque measured at the torsion bar. This results in a lower measured torque magnitude for the torsion bar. However, the overall pattern of applied torque is similar for the driver data and the experimental data.

Bibliography

[1] T. Dang, J. Desens, U. Franke, D. Gavrila, L. Schäfers, and W. Ziegler, "Steering and Evasion Assist," in *Handbook of Intelligent Vehicles*, vol. 1-2, pp. 1–1599, 2012.

[2] R. Bentley, *Speed secrets: professional race driving techniques*. 1998.

[3] "Model Predictive Control - MPC technology from ABB - What is new."

[4] B. Yi, *Integrated Planning and Control for Collision Avoidance Systems*. 2018.

[5] S. Singh, "Critical reasons for crashes investigated in the National Motor Vehicle Crash Causation Survey," *National Highway Traffic Safety Administration*, no. February, pp. 1–2, 2015.

[6] D. Lechner and G. Malaterre, "Emergency Manuever Experimentation Using a Driving Simulator," *Sae 910016*, 1991.

[7] S. Singh, "Driver attributes and rear-end crash involvement propensity," pp. 1–29, 2003.

[8] J. N. Dang, "Statistical Analysis of the Effectiveness of Electronic Stability Control (ESC) Systems Final Report," *NHTSA Report No. DOT HS*, vol. 810, no. July, p. 794, 2007.

[9] E. Camacho and C. Alba, *Model predictive control*. 2013.

[10] Bovag, "Mobiliteit in Cijfers Auto's 2015/2016," 2015.

[11] V. A. Laurense, J. Y. Goh, and J. C. Gerdes, "Path-tracking for autonomous vehicles at the limit of friction," *Proceedings of the American Control Conference*, pp. 5586–5591, 2017.

[12] 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.

[13] A. Katriniok and D. Abel, "LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics," *Proceedings of the IEEE Conference on Decision and Control*, pp. 6828–6833, 2011.

[14] N. R. Kapania, J. Subosits, and J. Christian Gerdes, "A Sequential Two-Step Algorithm for Fast Generation of Vehicle Racing Trajectories," *Journal of Dynamic Systems, Measurement, and Control*, vol. 138, no. 9, p. 091005, 2016.

[15] S. Heinrich, "Obstacle Avoidance with Safety Guarantees Feasibility of MPC-based Steering Algorithms," p. 95, 2015.

[16] S. J. Anderson, S. C. Peters, T. E. Pilutti, and K. Iagnemma, "Design and development of an optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios," *Springer Tracts in Advanced Robotics*, vol. 70, no. STAR, pp. 39–54, 2011.

[17] A. Carvalho, Y. Gao, A. Gray, H. E. Tseng, and F. Borrelli, "Predictive control of an autonomous ground vehicle using an iterative linearization approach," *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, no. Itsc, pp. 2335–2340, 2013.

[18] S. M. Erlien, S. Fujita, and J. C. Gerdes, "Shared Steering Control Using Safe Envelopes for Obstacle Avoidance and Vehicle Stability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 2, pp. 441–451, 2016.

[19] J. Funke, M. Brown, S. M. Erlien, and J. C. Gerdes, "Collision Avoidance and Stabilization for Autonomous Vehicles in Emergency Scenarios," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 4, pp. 1204–1216, 2017.

[20] Y. Gao, "Model Predictive Control for Autonomous and Semiautonomous Vehicles," *ProQuest Dissertations and Theses*, 2014.

[21] J. V. Frasch, A. Gray, M. Zanon, H. J. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," *Control Conference (ECC), 2013 European*, pp. 4136–4141, 2013.

[22] A. Gray, Y. Gao, T. Lin, J. K. Hedrick, H. E. Tseng, and F. Borrelli, "Predictive control for agile semi-autonomous ground vehicles using motion primitives," *American Control Conference (ACC), 2012*, pp. 4239–4244, 2012.

[23] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat, "Predictive active steering control for autonomous vehicle systems," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 3, pp. 566–580, 2007.

[24] H. B. P. . E. Bakker, "Experience with the magic formula tyre model," *Vehicle System Dynamics*, vol. 21, no. sup1, pp. 30–46, 1992.

[25] P. Falcone, H. Eric Tseng, F. Borrelli, J. Asgari, and D. Hrovat, "MPC-based yaw and lateral stabilisation via active front steering and braking," in *Vehicle System Dynamics*, vol. 46, pp. 611–628, 2008.

[26] V. Cars, "Volvo Cars and NVIDIA deepen ties," 2018.

[27] T. Shim, G. Adireddy, and H. Yuan, "Autonomous vehicle collision avoidance system using path planning and model-predictive-control-based active front steering and wheel torque control," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 226, no. 6, pp. 767–778, 2012.

[28] A. Wachter, *On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming.* 2005.

[29] 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*, 2018.

[30] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm forÂăquadratic programming," *Mathematical Programming Computation*, vol. 6, no. 4, pp. 327–363, 2014.

[31] B. Houska, H. J. Ferreau, and M. Diehl, "Automatica An auto-generated real-time iteration algorithm for nonlinear MPC in the," *Automatica*, vol. 47, no. 10, pp. 2279–2285, 2011.

[32] J. Andersson, "Performance of Casadi vs. Acado for NMPC [Online forum comment]," 2017.

[33] V. Leek, "An Optimal Control Toolbox for MATLAB Based on CasADi," 2016.

[34] J. Andersson and M. Diehl, *User Documentation for CasADi v2.2.0.* 2015.

[35] H. Bock and K. Plitt, "A Multiple Shooting Algorithm for Direct Solution of Optimal Control Problems *," *IFAC Proceedings Volumes*, vol. 17, no. 2, pp. 1603–1608, 1984.

[36] P. T. Boggs, W. J. Tolle, and J. W. Tolle, "Sequential Quadratic Programming," *Acta Numerica*, vol. 4, no. November 2008, pp. 1–52, 1996.

[37] J. M. Maciejowski, "(Maciejowsky, 2000) - Predictive Control with Constraints," 2002.

[38] M. Diehl, H. G. Bock, H. Diedam, P.-b. Wieber, M. Diehl, H. G. Bock, H. Diedam, P.-b. W. Fast, and D. Multiple, "Fast Direct Multiple Shooting Algorithms for Optimal Robot Control," 2009.

[39] J. T. Betts, "Survey of Numerical Methods for Trajectory Optimization," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[40] D. Yang, T. J. Gordon, B. Jacobson, and M. Jonasson, "Quasi-linear optimal path controller applied to post impact vehicle dynamics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 4, pp. 1586–1598, 2012.

[41] D. I. Katzourakis, *Driver Steering Support Interfaces Near the Vehicle's Handling Limits.* 2012.

[42] Y. Gao, A. Gray, J. V. Frasch, T. Lin, E. Tseng, J. K. Hedrick, and F. Borrelli, "Spatial Predictive Control for Agile Semi-Autonomous Ground Vehicles," *Proceedings of the 11th International Symposium on Advanced Vehicle Control*, vol. VD11, no. 2, pp. 1–6, 2012.

[43] H. B. Pacejka, *Tyre characteristics and vehicle handling and stability.* 2007.

Glossary

List of Acronyms

MPC	Model Predictive Control
NMPC	Nonlinear Model Predictive Control
PID	Proportional-Integral-Derivative control
NLP	Nonlinear Programme
QP	Quadratic Programming
IPM	Interior-Point Method
SQP	Sequential Quadratic Programming
KKT	Karush-Kuhn-Tucker Conditions
EMA	Evasive Manoeuvre Assistance Function
IMU	Inertial Measurement Unit
RTI	Real-time Interface
ADAS	Advanced Driver Assistance Systems
CAN	Controller Area Network
ECU	Electronic Control Unit
DSR	Driver Steering Recommendation
ESC	Electronic Stability Control

List of Symbols

B_d	Driveline damping coefficient
H	Time horizon
h	Control interval time step
I_z	Yaw moment of inertia of the vehicle C_1
I_{d_i}	Wheel moment of inertia C_2
m	Mass of the vehicle
N	Horizon
r	Wheel radius
T_{b_i}	Brake torque at wheel C_2
u_r	Reference control input
w	Track width
x_r	Reference state vector
X_s	Longitudinal distance travelled to reach $Y = 2\,\mathrm{m}$
ω_i	Angular velocity of wheel C_2
C_0	Global coordinate frame
C_1	Local vehicle coordinate frame
C_2	Local wheel coordinate frame
l_f	Distance from centre of gravity to front axle
l_r	Distance from centre of gravity to front axle
α_i	Slip angle C_2
$\ddot{x}$	Longitudinal acceleration C_1
$\ddot{y}$	Lateral acceleration C_1
$\dot{\delta}$	Steering rate C_1
$\dot{\psi}$	Yaw rate C_1
$\dot{x}$	Longitudinal velocity C_1
$\dot{y}$	Lateral velocity C_1
κ	Slip ratio
F_{c_i}	Lateral tyre force C_2
F_{l_i}	Longitudinal tyre force C_2
F_{z_i}	Normal force on wheel
u	Control input
v_{c_i}	Lateral wheel velocity C_2
v_{l_i}	Longitudinal wheel velocity C_2
v_{x_i}	Longitudinal wheel velocity C_1
v_{y_i}	Lateral wheel velocity C_1
x	State vector
β	Body slip angle
δ	Steering angle C_1

ψ	Yaw angle C_1
X	Global longitudinal position C_0
Y	Global lateral position C_0