



Peter Reithuber, BSc

**Development and Simulation of a Vehicle MPC Controller  
using an Advanced Vehicle Dynamics Model of a Four-Wheel  
Driving/Steering Electric Vehicle**

**MASTER THESIS**

to achieve the university degree of

Master of Science

Master degree programme: Mechanical Engineering

submitted to

**Graz University of Technology**

Supervisor

Assoc.Prof. Dipl.-Ing. Dr.techn. Mario Hirz

Institute of Automotive Engineering

Prof. Dr.-Ing. Su Zhou

School of Automotive Studies, Tongji University Shanghai



## **AFFIDAVIT**

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material, which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

---

Date

---

Signature



## Abstract

Autonomous driving is a very current topic and both vehicle manufacturers and researchers around the world work intensively in that field. The School of Automotive Studies at Tongji University Shanghai has developed a four-wheel independent drive, four-wheel independent steering (4WID-4WIS) vehicle, which is the basis for research around the topic of autonomous driving. This thesis deals with one of many topics situated in the field of vehicle control by using Model Predictive Control (MPC). In the beginning, the basic research goal and motivation in general and specific context, which is related to the 4WID-4WIS vehicle, is introduced. The prototype is described briefly in order to give an idea of how the special vehicle architecture is realized. This introduction is followed by a closer look on Model Predictive Control, its basics, working principle, advantages and disadvantages. Building on this, a possible control hierarchy for autonomous vehicles is introduced and discussed. This includes the description and definition of special driving modes, which the vehicle is able to perform thanks to its advanced vehicle architecture. Additionally, the different levels of the control hierarchy are lined out and described. Having introduced the basic idea behind Model Predictive Control and the control architecture, the design process of the vehicle controllers is described. There, the first steps include a mathematical description of the prediction models used by the MPC controllers, which are transformed in a way that they are suitable for the use in MATLAB<sup>®</sup>. In the end, a control structure is presented, which includes a MPC Motion Planner, a MPC Steering Controller and a MPC Torque Vectoring Controller. After the controller design, the implementation in MATLAB<sup>®</sup> and Simulink<sup>®</sup> is taken care of, during which several important steps in controller tuning and evaluation are described. These steps ensure proper control behavior and the corresponding section of the thesis provides information about the control stability and robustness. The second to last chapter describes a big variety of conducted simulations, like the evaluation of different MPC strategies with respect to the influence of varying vehicle speed or a comparison between a PID steering controller and a MPC steering controller. Furthermore, the advantages of four-wheel steering are assessed and the influence of torque vectoring is described. In addition, this section includes information about the special modes, which are simulated in order to evaluate the abilities of the MPC control architecture to handle these situations. The chapter is concluded by the simulation of combined driving modes, which complete driving tasks like sideways parking, turning around in a dead end or performing a right angle turn on the spot. In the last chapter, a conclusion of the conducted work is drawn and several possible future research topics following and advancing the presented control approach are listed.

**Key Words:** Model Predictive Control, autonomous driving, four-wheel steering, driving modes



# Table of Contents

<b>1</b>	<b>Introduction</b> .....	<b>1</b>
1.1	Autonomous Driving in Thesis Context .....	1
1.2	Current Research Status and Motivation .....	2
1.3	The 4WID-4WIS Electric Vehicle .....	3
1.4	Main Research Content.....	4
<b>2</b>	<b>Model Predictive Vehicle Control</b> .....	<b>5</b>
2.1	Basic Principle and Background.....	5
2.2	MPC Structure and Mathematical Formulation.....	7
<b>3</b>	<b>Control Architecture</b> .....	<b>10</b>
3.1	Control Hierarchy .....	10
3.2	Navigation Level and Driving Modes.....	11
3.2.1	Four-Wheel Steering Mode .....	11
3.2.2	Sideways Driving .....	13
3.2.3	Turning on the Spot.....	14
3.2.4	Overview of Driving Modes and Decision Making .....	14
3.3	Motion Planning and Obstacle Avoidance.....	16
3.4	Trajectory Following.....	17
<b>4</b>	<b>Prediction Models for Controller Design</b> .....	<b>18</b>
4.1	Motion Planner .....	19
4.1.1	Kinematic Bicycle Model for Motion Planning .....	19
4.2	Steering Controller .....	21
4.2.1	Dynamic Bicycle Model for Steering Control .....	22
4.2.2	Ackermann Four-Wheel Steering.....	25
4.3	Torque Vectoring Controller .....	27
4.3.1	Torque Vectoring Principle .....	28
4.3.2	Extended Dynamic Bicycle Model for Torque Vectoring .....	31
4.4	Complete Control Structure.....	35
<b>5</b>	<b>MPC Controller Design in MATLAB® and Simulink®</b> .....	<b>37</b>
5.1	Controller Design in MATLAB® Command Line .....	37
5.2	MPC Controller Tuning.....	38
5.3	MPC Block in Simulink® .....	42
5.4	Adaptive MPC Block in Simulink® .....	43
5.5	Kinematic Steering in Simulink® .....	45
<b>6</b>	<b>Simulations</b> .....	<b>46</b>

<b>6.1</b>	<b>Advanced Vehicle Model used for Simulation</b> .....	46
<b>6.2</b>	<b>Simulation Scenarios</b> .....	46
<b>6.3</b>	<b>Simulation of MPC Steering Controller</b> .....	47
6.3.1	Double Lane Change with Normal MPC .....	47
6.3.2	Double Lane Change with Adaptive MPC .....	50
6.3.3	Double Lane Change with Positive and Negative Steering .....	54
<b>6.4</b>	<b>Simulation of a PID Steering Controller</b> .....	55
6.4.1	Design of PID Steering Controller .....	55
6.4.2	Comparison with MPC Steering Controller and Conclusion .....	56
<b>6.5</b>	<b>Simulation of MPC Torque Vectoring Controller</b> .....	60
6.5.1	Double Lane Change with and without Torque Vectoring .....	60
6.5.2	Sideways Driving .....	62
6.5.3	Turning on the Spot .....	64
<b>6.6</b>	<b>Simulation of Obstacle Avoidance</b> .....	65
6.6.1	Obstacle Avoidance Scenario .....	65
6.6.2	Implementation in Simulink® .....	66
6.6.3	Results and Discussion .....	66
<b>6.7</b>	<b>Simulation of Combined Modes</b> .....	67
6.7.1	Parking Scenario .....	67
6.7.2	Dead-End Scenario .....	68
6.7.3	Right Angle Turn .....	69
<b>7</b>	<b>Conclusion and Outlook</b> .....	71
<b>7.1</b>	<b>Conclusion</b> .....	71
<b>7.2</b>	<b>The Direction of Further Work</b> .....	71
<b>8</b>	<b>References</b> .....	72
<b>9</b>	<b>Appendix</b> .....	77

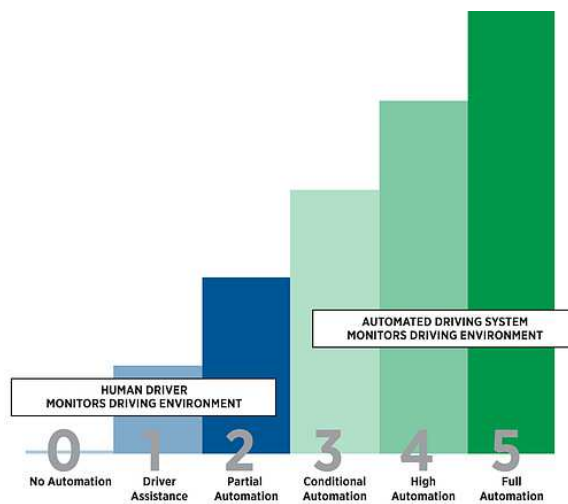


# 1 Introduction

## 1.1 Autonomous Driving in Thesis Context

Autonomous driving is a major field of research in both the automotive industry and at the academic level and concerns not only the scientific community but also politics and customers. One of the big potentials of autonomous driving is seen to be in reducing fatal car accidents by ruling out the human driver as the source of errors [1]. It also promises advantages in passenger and goods transportation.

Automated driving can be subdivided into several different levels, as described by SAE [2]. Figure 1-1 shows these five levels.



**Figure 1-1** SAE Levels of Automation

Level “0” refers to a conventional car, which is solely controlled by a human driver. Level “1” means that the vehicle is able to control the vehicle longitudinal dynamics to a certain extent, whereas Level “2” takes both longitudinal and lateral dynamics into account. The systems of the vehicle play in these levels an assisting role and the driver still has to maintain supervision over these functions and in case take over control again. In the highly automated Level “3”, the car is able to perform certain tasks and manoeuvres by itself during the driver can do other things. But he/she is still required to take over the controls in case the vehicle can’t handle a situation or the system fails. In the fully automated Level “4” no human intervention is needed to perform the whole driving task. But there is still a possibility for the driver to take over the controls if necessary. At the final autonomous Level “5”, the vehicle moves absolutely independent of any human interaction. The vehicle, therefore, turns into a self-driving robot, which can operate on its own [3].

This thesis is part of ongoing research at the School of Automotive Studies at Tongji University, which deals with the automation of a highly maneuverable electric car, introduced in Chapter 1.3. The pursued level of automation is level 5, according to SAE/VDA. Considering the abilities, the vehicle should have, some basic guidelines were defined, which for example include the field of application and research goals. As the field of application, inner-city traffic or transport is taken into account, which means that the vehicle will travel at low speeds. In order to be suitable for numerous transportation tasks, including persons and goods, the

vehicle should be able to manoeuvre in confined space by utilizing its high level of maneuverability and therefore represent a difference to conventional vehicles in these fields of application.

## 1.2 Current Research Status and Motivation

A vehicle with four-wheel independent drive and steering is a distributed nonlinear control system [4] and therefore requires a special control approach, which is able to handle a system of that kind. In the field of vehicle control, there are many different approaches in use. [5] gives an overview of the broad topic of path tracking, like for example strategies based on a geometric or kinematic representation of the vehicle, which are simple and have computational advantages but fail to represent the dynamic behavior of the vehicle. This is not the case with dynamic vehicle models, which take the moments and forces acting on the vehicle into account and are therefore able to represent the vehicle in a proper way - even when driving at the limits of vehicle dynamics. As mentioned before, the vehicle is a nonlinear system and this is mainly caused by the forces, which act between the tires and the road [6]. These forces are highly nonlinear and there are models available, which help to imitate this behavior for control design, like the "Magic Formula" from Pacejka [7], which is commonly used for designing nonlinear controllers. Since using a nonlinear controller rises complexity in both controller design and computational aspects, it is common to linearize the tire models by accepting the loss of a more accurate vehicle representation [5].

The following list should give a brief overview of control strategies, which are used in the field of four-wheel steering control design considering nonlinear controlled systems. A control type, which is suitable for controlling nonlinear and linear systems, is the Fuzzy Control and is applied in many different fields of application. The Fuzzy Controller is a purely statistic mapping controller without dynamics. Under some effort in optimizing Fuzzy Control, a behavior similar to a human can be achieved for autonomous driving [8]. Another approach is described in [9] where a Backstepping Control algorithm is established based on a kinematic vehicle model in order to control a nonlinear four-wheel steering vehicle for trajectory tracking. It was successfully simulated and tested on a prototype following trajectories with sharp turns. The Sliding Mode Controller (SMC) uses the control signals as discontinuous functions and is therefore suitable for controlling nonlinear systems. It offers a control strategy, which responds fast and is highly robust against external disturbances. But due to the high switching frequency of the controller, the signals tend to chatter and that could raise problems with actuators, which can't follow the rapid changing signals or even be damaged [5]. In [10]  $H_\infty$  control and in [11]  $H_2/H_\infty$  mixed robust control are used for the steering control. These control types combine robust performance and robust stability and show good results in vehicle stability control.

A very promising approach for controlling nonlinear systems is Model Predictive Control (MPC) because it can handle multiple-input multiple-output systems (MIMO) in which the signals can have interactions between their inputs and outputs. It can also control several variables simultaneously, allows interactions between several input and output signals and handles constraints on signal values and the signal rate-of-change. Besides these characteristics, it offers a preview capability, which makes it especially interesting for applications in autonomous driving since it offers a similar behavior like a human driver by

predicting future control inputs using a prediction model of the controlled system. These capabilities are the main motivation why the MPC control is used in this thesis. A drawback, however, is that Model Predictive Control is computationally complex and therefore requires appropriate controller hardware. The details of Model Predictive Control will be described and discussed throughout this thesis.

### 1.3 The 4WID-4WIS Electric Vehicle

A research group at the School of Automotive Studies at Tongji University has designed and built a prototype of a battery electric vehicle with four individually drive- and steerable wheels. The 4WID-4WIS (Four-Wheel Driving Four-Wheel Steering) electric vehicle serves the purpose of research work in the fields of vehicle chassis control and autonomous driving and represents the basis for the work conducted in this thesis.



**Figure 1-2** The 4WID-4WIS Vehicle Prototype

The special design can be seen in Figure 1-2, with four individual wheel-hub motors and a steering motor at each wheel, positioned above the wheels. With this vehicle architecture, a higher maneuverability can be achieved compared to a common two-wheel steering vehicle since all four wheels can be steered and driven individually. That allows for special driving manoeuvres or driving modes. Those will be discussed in Chapter 3.2. Table 1 shows selected parameters of the 4WID-4WIS vehicle, which are considered in controller design.

**Table 1** 4WID-4WIS vehicle parameters

Full mass	$m$	500 kg
Tire radius	$r_w$	0,2521 m
Length of vehicle	$L$	2,2 m
Width of vehicle	$B$	1,4 m
Distance CoG to front axle	$L_f$	1,05 m
Distance CoG to rear axle	$L_r$	1,15 m
Vehicle Moment of Inertia about z-axis	$I_z$	488 kgm <sup>2</sup>
Wheel Moment of Inertia	$I_w$	1,5 kgm <sup>2</sup>

## **1.4 Main Research Content**

The vehicle prototype, which is introduced in Chapter 1.3 is a research vehicle in the field of autonomous driving. One crucial aspect of autonomous driving is the chassis control, which has to process data from a navigation system and compute driving commands by bearing the vehicle dynamics and physical limitations in mind. The main research content of this thesis aims at providing chassis control based on MPC control, which is suitable for the 4WID-4WIS vehicle. This vehicle is designed to be an autonomous vehicle and the control concept of this thesis should, therefore, be suitable for this field of application. That is why a motion planner for computing the vehicle trajectory and enabling obstacle avoidance should be designed and simulated as well. Furthermore, new driving functions should be evaluated, implemented and simulated by utilizing the four-wheel independent steering system.

## 2 Model Predictive Vehicle Control

### 2.1 Basic Principle and Background

The roots of Model Predictive Control date back to 1979 when the concept was first introduced in the petrochemical industry sector. With the new approach of predicting system behavior, it was shown that the control outputs were less aggressive and led to a better convergence with the reference values. In the beginning, the predictive control strategy was only used in slow processes where computation time was not crucial because the optimization is computationally complex. Only with the development of faster solvers and more capable controller hardware, it was possible to use MPC in highly dynamic systems like e.g. vehicles [12].

Model Predictive Control is an interesting control approach for autonomous vehicle control because of the predictive nature, which is similar to human behavior. A driver will look ahead on the road to monitor its environment and will, therefore, be able to react in time to both sudden and predictable events or changes in the driving path due to e.g. turns or overtaking manoeuvres. Model Predictive Control makes it possible to realize a similar behavior and is therefore in combination with a sufficient perception system very suitable for autonomous driving [12].

When talking about the basics of Model Predictive Control it makes sense to first have a look at Optimal Control to understand the working principle of MPC and the benefits of its algorithm. The Optimal Control strategy is to determine a set of control inputs to a dynamical system in such a way that the performance index  $J$  is minimized. This optimization is subject to initial conditions, constraints and system dynamics and is performed over a defined time horizon. The performance index  $J$ , which should be minimized, could be formulated as follows:

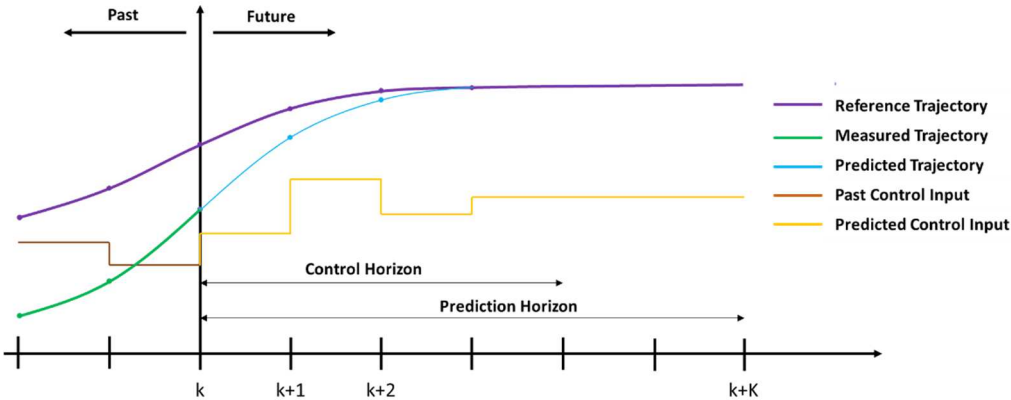
$$J = \vartheta[x(K)] + \sum_{k=0}^{K-1} \varphi(x(k+1), \mathbf{u}(k), \mathbf{d}(k)) \quad (2.1)$$

Equation (2.1) shows the performance index  $J$ , where  $\vartheta$  is the terminal cost,  $K$  is the time horizon,  $\varphi$  is the Lagrangian cost,  $x$  is the state vector,  $\mathbf{u}$  is the vector of manipulated control inputs and  $\mathbf{d}$  is the disturbance vector. This control strategy allows for optimal controller outputs over the considered time horizon, which are kept within a permissible range thanks to constraints.

But the Optimal Control method has a major drawback because it basically represents an open-loop control. Due to this, crucial errors are not taken into account like e.g. model mismatch or external disturbances and the effects of those can therefore not be lowered by the controller [13].

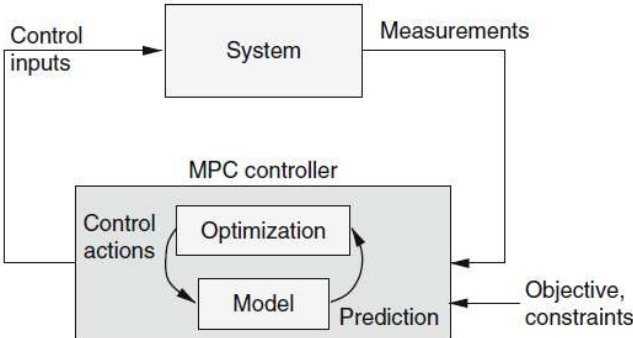
The strategy of MPC aims at optimizing a predicted cost, which will give the future control inputs. These control inputs are themselves limited by constraints in order to gain permissible values and only the first input resulting from the optimization is fed to the system. At the next time step, the whole process is repeated with newly updated information about the system states. That enables the MPC to respond to deviations between control outputs and actual system response and therefore gain a certain robustness against uncertainties like e.g. prediction model mismatch or disturbances [14].

Model Predictive Control is related to Optimal Control to a certain degree since it performs a similar optimization, but does that repeatedly only for a limited time span known as the prediction horizon  $N_p$ . This approach is also known as rolling or receding horizon and gives MPC the characteristics of a feedback controller. As mentioned before, the computational complexity is a limiting factor for MPC applications and in order to reduce the needed computing power, the control horizon  $N_c$  is introduced, which is smaller than the prediction horizon. The basic optimization is not changed by this but it will only be performed until the end of the control horizon. After that, the control input is kept constant until the end of the prediction horizon. The introduction of  $N_c$  does not only help to reduce computational complexity but also improves the stability of the controller. Another advantage of the rolling horizon routine is that it results in an online adaptive control scheme and this makes it possible to incorporate certain changes of e.g. system inputs by updating the prediction model [13]. This advantage will be used in the thesis and is described in Chapter 5.4.



**Figure 2-1** Prediction of control inputs during the control interval

Figure 2-1 shows past and future control inputs over time, control step  $k$ , number of steps  $K$ , control horizon  $N_c$ , prediction horizon  $N_p$  as well as reference, measured and predicted trajectory.



**Figure 2-2** Control loops of MPC [13]

Figure 2-2 [13] shows the two loops, which are performed in the arrangement of the MPC with the controlled system. The optimization loop within the controller repeats several times until sufficiently good control signals are found. The outside loop, on the other hand, will only be performed once at each control step  $k$ . It connects the system with the controller and also gives the feedback signals to the controller [13].

## 2.2 MPC Structure and Mathematical Formulation

This chapter aims at giving a general overview of the structure of MPC and the basic mathematical description. Since the development and simulation of the controllers used in this thesis was done in MATLAB® using predefined functions to create MPC, the optimization problem will not be formulated explicitly for each controller. Therefore, this chapter will only show an exemplary mathematical formulation.

### System Model

In Model Predictive Control the prediction model plays an important role. It is derived from the actual controlled system and simplified in a way to balance between a sufficiently accurate representation of the system behavior and simplicity in order to keep the computational complexity low. In case of a vehicle, the system itself is highly nonlinear, mostly caused by external forces on the vehicle, namely the tire forces [6]. MPC is able to control nonlinear systems and is, therefore, suitable for the use in the automotive field. Considering the formulation of system models there are basically two approaches, which are either using a nonlinear model or a linear model and that also asks for either nonlinear or linear Model Predictive Controllers. The advantage of linear MPC over nonlinear MPC is the lower computational complexity, whereas the nonlinear MPC can represent the system dynamics better and is, therefore, more suitable for a wider range of applications. In case of vehicle control this would be driving at high vehicle dynamics or even at the limits [15]. Another advantage of nonlinear MPC is that the stability boundary of the controlled system can be raised [6]. As mentioned in Chapter 1.1, the field of application considered in this thesis is limited to inner-city driving and maneuvering at low speeds and low vehicle dynamics. Therefore, and due to the advantage of low computational complexity, only linear prediction models and linear MPC are considered further on.

In this chapter, a general mathematical formulation of the Model Predictive Control optimization problem is shown, based on state space formulation since this is used further on throughout Chapter 4. Equations (2.2) and (2.3) show the general state space representation where  $x$ ,  $y$  and  $u$  are the state, input and output matrices respectively and  $A$ ,  $B$ ,  $C$  and  $D$  are the system matrices.

$$\dot{x} = Ax + Bu \quad (2.2)$$

$$y = Cx + Du \quad (2.3)$$

In order to match with the finite dimensional optimization problem of MPC, Equations (2.2) and (2.3) are displayed in discrete state space form. For doing that, the following formulation is derived from the multiple-input multiple-output (MIMO) system used in [16] and adapted to the notation used in this thesis.

$$x(k+1) = A_d x(k) + B_d u(k) \quad (2.4)$$

$$y(k) = C_d x(k) + D_d u(k) \quad (2.5)$$

Equations (2.4) and (2.5) show the discretized state space model, where  $x(k)$  is the state variable,  $y(k)$  represents the output of the model and  $u(k)$  the input.  $A_d = A \cdot T_s$ ,  $B_d = B \cdot T_s$ ,  $C_d = C$  and  $D_d = D$  are the discretized system matrixes with the sampling time  $T_s$ . Furthermore,

this formulation contains no dead time since the MPC Controller in MATLAB compensates that [17].

A new state vector  $\xi(k)$  is now introduced, which is composed of  $x(k)$  and  $u(k-1)$  and is displayed in vector form:

$$\xi(k) = \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} \quad (2.6)$$

Implementing the new state vector  $\xi(k)$  in the discretized state space form leads to:

$$\xi(k+1) = \widetilde{A}_d \xi(k) + \widetilde{B}_d \Delta u(k) \quad (2.7)$$

$$y(k) = \widetilde{C}_d \xi(k) + \widetilde{D}_d \Delta u(k) \quad (2.8)$$

The matrixes  $\widetilde{A}_d$ ,  $\widetilde{B}_d$  and  $\widetilde{C}_d$  are composed of the discretized system matrixes described before and  $\widetilde{D}_d = D_d$ . The structure of these matrixes depends on the considered system and will, therefore, be left out in this general description.

### Cost Function

With the optimal control input vector  $\Delta u(k+i|k)$  a cost function is formulated. The objective of Model Predictive Control is to minimize this cost function.

$$\min_{\Delta U(k)} \left[ J(\xi(k), \Delta U(k)) = \sum_{i=1}^{N_p} \|y(k+i|k) - y_r(k+i|k)\|_Q^2 + \sum_{i=0}^{N_c-1} \|\Delta u(k+i|k)\|_R^2 \right] \quad (2.9)$$

In Equation (2.9)  $N_c$  and  $N_p$  are the control and prediction horizon respectively,  $y_r$  is the reference for the controller. The diagonal Matrixes  $Q$  and  $R$  represent the weighting matrices.

Furthermore, the equation is subject to Equation (2.10) and the system constraints.

$$\xi(k+i|k) = \widetilde{A}_d \xi(k+i-1|k) + \widetilde{B}_d \Delta u(k+i-1|k) \quad (2.10)$$

### Constraints

One of the advantages of a Model Predictive Controller is that the system constraints can be defined in the cost function and are therefore taken into account in the optimization. This makes it possible to incorporate system constraints, which might be necessary because of physical limitations or to represent boundaries, which should not be violated due to safety reasons. Constraints are formulated as inequalities to define a range within the values can vary and also on the signal rate-of-change. They can be applied to both input and output values and have the basic structure as in Equation (2.11) and (2.12).

$$y_{min} \leq y(k+i|k) \leq y_{max} \quad (i = 1, 2, \dots, N_p) \quad (2.11)$$

$$u_{min} \leq u(k+i|k) \leq u_{max} \quad (i = 0, 1, \dots, N_c - 1) \quad (2.12)$$

### Optimizer

There are several different solvers available for the optimization process within the MPC framework. When choosing a solver, it is necessary to know if the optimization problem is convex or non-convex. In mathematical terms, convexity is given for a range of functions and sets if the connecting line between two points of the set lies entirely within the set. For an optimization, it is crucial to have convex sets because only by that it can be guaranteed to get



a global optimum in the solution. The model for a convex optimization problem must be linear, otherwise, special solver methods need to be applied, which would make the online optimization computationally more complex. In other words, for the use of linear MPC the model must also be linear and by that, a convex optimization can be used, which results in an optimal solution [12]. Furthermore, when the problem formulation is convex, linear and the cost function is quadratic it's called a quadratic program (QP). On the other hand, if the model is nonlinear it is in general also non-convex and instead of a global optimum, only a local optimum can be achieved [18].

In this thesis, the MPC controllers are designed in MATLAB® using predefined functions and the standard KWIK algorithm [19] to solve the quadratic program (QP) problem.

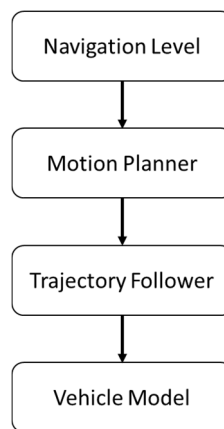
### Controller Stability

In controller development, it is important to prove stability in order to gain reliable and safe working behavior in practical applications. The control law of MPC is nonlinear thanks to the constraints, which are described by inequalities; therefore, Lyapunov stability theory can be applied. This is under the assumption that there is no model mismatch and that all disturbances can be modeled and therefore compensated [14], [18]. In order to keep the computational complexity low, however, it is common practice to evaluate and ensure the stability by conducting a series of simulations and experiments [20], [16]. Since in this thesis the controller is developed in MATLAB®, a possible way of checking controller stability will be shown in Chapter 5.2.

## 3 Control Architecture

### 3.1 Control Hierarchy

Depending on the level of automation, an autonomous vehicle needs to fulfill a certain amount of tasks, which would normally be done by the driver, as described in Chapter 1.1. This extends to a level where all driving tasks are fulfilled automatically by the vehicle without any human action. In order to realize this ability, several measures have to be taken in order to drive the vehicle safely. As described in [1] these measures can be split up into a hierarchy beginning with route planning at the top, followed by behavioral decision making, motion planning and vehicle control. This and similar basic structures are widely used in e.g. [20], [21], [22] and [23] and will also provide the basis for the control architecture in this thesis.



**Figure 3-1** Control Hierarchy of Autonomous Vehicles

As shown in Figure 3-1 a navigation level represents the top of the hierarchy and provides data for the motion planner, which itself gives commands to the lower level trajectory follower. The feedback flow from the bottom to the top of the hierarchy is left out in this depiction. The base of the structure is represented by the controlled system, which can be a vehicle or as in case of this thesis a vehicle model. In the next chapters, the different levels of the hierarchy will be discussed.

Splitting up control levels as shown above also has practical reasons, since they perform different tasks, which also ask for different parameters and settings within the controllers, e.g. the sampling time. Another aspect is that not all levels need the same information and a control architecture can help to channel and distribute the signals and data accordingly. It should be mentioned here, that the main focus of this thesis lies on the levels of motion planning and trajectory following, rather than on the navigation level or the vehicle model. Both motion planning and trajectory following are realized using individual Model Predictive Controllers. The strategy of using two different controllers for these levels has certain advantages since the controllers can be tuned and optimized separately, specifically for their task. Therefore, things like prediction horizon and sampling time can be chosen individually, which e.g. allows the motion planner to have a long prediction horizon in order to take many obstacles into account and to perform the optimization at low frequency. The trajectory follower, on the other hand, can operate with a shorter prediction horizon and a shorter sampling time. Therefore, it

can cope better with fast signals from the vehicle and respond quickly to sudden changes but over a smaller time horizon, which results in a computational advantage [20], [21].

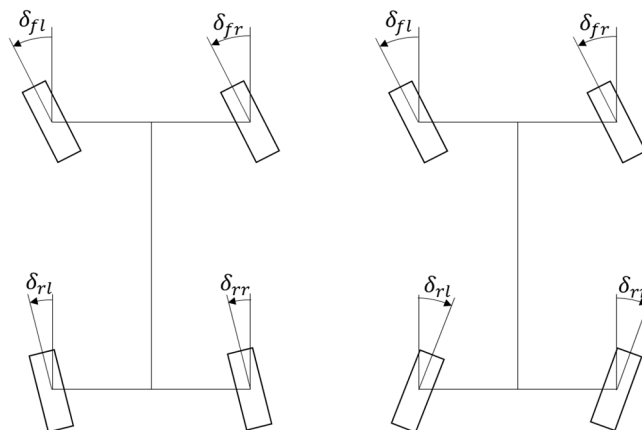
### 3.2 Navigation Level and Driving Modes

The navigation level is a crucial part of the control framework of an autonomous vehicle because it represents the link between the environment surrounding the vehicle and all lower control levels. An example includes map data that has to be available from which a coarse path can be planned in the vehicle environment. In order to do that, the vehicle must be able to localize itself either by the perception system, other positioning systems or even communication with other vehicles (V2V) or to infrastructure (V2I) [23]. The perception system is composed of several different sensors and sensor systems like cameras, radar, LIDAR, GPS and several low-range sensors [1]. Combining all these functionalities in the navigation level represents one of the prior goals of autonomous driving that must be ensured at any time, which is providing safety for both the environment of the vehicle and the passengers [23].

The Navigation Level and its functionality is not the primary concern of this thesis, but it delivers certain information for the lower level controllers. This includes the planned path and a velocity profile. Thanks to the special architecture of the 4WID-4WIS vehicle, several different driving modes can be realized. These driving modes enable the vehicle to perform manoeuvres, which would not be possible with a conventional vehicle. In the following sections, the driving modes are described and distinguished by the values of longitudinal velocity, lateral velocity and yaw rate in the vehicle coordinate system.

#### 3.2.1 Four-Wheel Steering Mode

This driving mode utilizes one of the big advantages of four-wheel steering considering vehicle dynamics. Four-wheel steering can be distinguished in two cases, one is positive steering and the other one is negative steering. The difference lies in the orientation of the rear wheels with respect to the front wheels.



**Figure 3-2** Positive and negative steering configuration

The left side of Figure 3-2 shows positive steering, where the rear wheels point in the same direction as the front wheels. The vehicle on the right shows negative steering, which is characterized by rear wheel steering angles pointing in the opposite direction of the front wheels [24].

Positive Steering

In this steering configuration, the steering wheels at the rear axle turn into the same direction as the ones at the front axle. This can improve stability and steering response at higher speeds [24]. Furthermore, simulations, which will be described in Chapter 6.3.3 have shown, that a vehicle yaw rate of almost zero can be achieved with this steering configuration. A drawback can be seen when evaluating the turning radius of the vehicle, since it is larger than the one achieved with a two-wheel steering vehicle and thus results in lower maneuverability considering e.g. narrow turns [25].

Negative Steering

As described above this steering configuration is characterized by the different turning direction of front and rear wheels. This means e.g. that the rear wheels will turn clockwise if the front wheels turn anticlockwise and vice versa. The main advantage of this steering configuration is that the turning radius of the vehicle can be considerably reduced compared to a conventional two-wheel steering vehicle and thus makes it suitable for vehicles, which have to provide high maneuverability. [25]

Table 2 lists the values of vehicle speed and yaw rate with which positive and negative steering manoeuver can be characterized. For positive steering, the yaw rate is set zero. This decision is described closer in Chapter 6.3.3.

**Table 2** Vehicle velocity signals of Four-Wheel Driving Mode

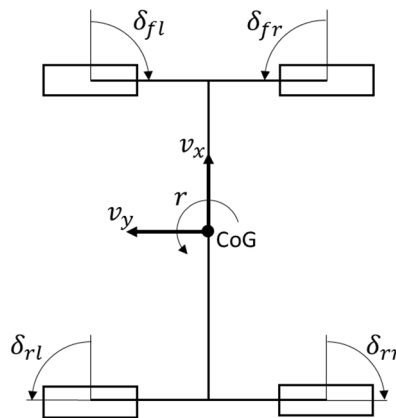
	Positive Steering	Negative Steering
Vehicle Longitudinal Speed	$-v_x \leq 0 \leq v_x$	$-v_x \leq 0 \leq v_x$
Vehicle Lateral Speed	$-v_y \leq 0 \leq v_y$	$-v_y \leq 0 \leq v_y$
Vehicle Yaw Rate	$r = 0$	$-r \leq 0 \leq r$

In this thesis only forward driving in positive x-direction in vehicle coordinate frame is considered, but backward driving with four-wheel steering configuration could be achieved the same way. In fact, the vehicle prototype shown in Chapter 1.3 is designed in a way, that there is no defined front or rear end of the vehicle. So the difference between forward and backward driving comes down to a change in signs of the driving inputs and doesn't ask for major changes.

### 3.2.2 Sideways Driving

Thanks to the special design of the 4WID-4WIS electric vehicle the wheels can be turned up to a 90 degrees angle, which allows driving in lateral vehicle direction. This driving manoeuver is called sideways driving further on.

Sideways driving is an interesting ability for many real-world applications. For example, it can be used to park sideways in a spot, which is only slightly longer than the vehicle itself. Such a manoeuver would not be possible with conventional steering. As a consequence of this ability, the necessary parking area for vehicles with such a steering configuration can be kept smaller than for conventional vehicles. This could result in a better utilization of available parking space in crowded cities. An example of such a parking scenario is shown in Chapter 6.7.1. Another application for sideways driving is to utilize the improved maneuverability for delivery tasks. A vehicle with the ability of lateral driving can be used in narrow streets where conventional steering isn't sufficient.



**Figure 3-3** Sideways Driving Steering Configuration

Figure 3-3 shows the steering configuration for sideways driving. The wheels are turned 90 degrees in the clockwise and counterclockwise direction respectively. Table 3 lists the values of vehicle speed and yaw rate with which the sideways driving manoeuver can be characterized.

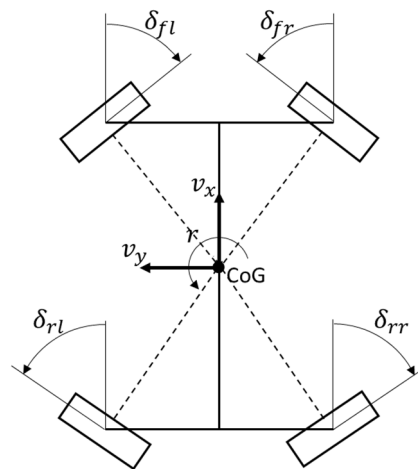
**Table 3** Vehicle velocity signals for Sideways Driving

Vehicle Longitudinal Speed	$v_x = 0$
Vehicle Lateral Speed	$-v_y \leq 0 \leq v_y$
Vehicle Yaw Rate	$r = 0$

It is obvious that this steering configuration can only be initialized when the vehicle stands still. In case the vehicle is traveling at a certain speed, a sudden 90 degrees turn of the wheels would cause the vehicle to lose stability and move uncontrolled. Therefore, a change into sideways driving mode can only be performed safely when the vehicle is standing still. It is important to consider this necessity in the planning stage of the navigation level.

### 3.2.3 Turning on the Spot

The vehicle architecture of the 4WID-4WIS vehicle described in Chapter 1.3 allows turning the steering wheels in a way that the vehicle will turn on the spot. In order to stay at the same position during the whole manoeuvre in any direction, both vehicle longitudinal and lateral speed have to be zero and the center of rotation should lie on the vehicle center of gravity (CoG), as it can be seen in Figure 3-4. This configuration allows the vehicle to be highly maneuverable since it can change its orientation right in the place where it is at the moment, which is not possible for a two-wheel steering vehicle. Therefore, it can e.g. enter dead-end streets, which are too narrow to turn the vehicle around with common two-wheel steering. In that case, the vehicle can turn on the spot and leave the dead-end street. Examples for using the turning on the spot mode are displayed in Chapter 6.7.2.



**Figure 3-4** Turning on the Spot Configuration

Table 4 shows vehicle speed and yaw rate for turning on the spot.

**Table 4** Vehicle velocity signals for Turning on the Spot

Vehicle Longitudinal Speed	$v_x = 0$
Vehicle Lateral Speed	$v_y = 0$
Vehicle Yaw Rate	$-r \leq 0 \leq r$

Similar to sideways driving this mode requests the setting of defined steering angles at the beginning of the manoeuvre. The steering angles are far bigger than normal and in order to avoid the vehicle to lose stability or drive uncontrolled, the steering angles can only be set when the vehicle speed is zero. This limitation has to be taken into account during route planning in the navigation level.

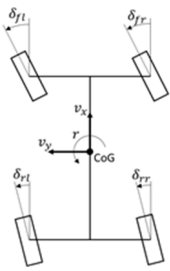
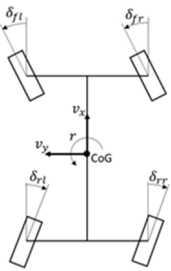
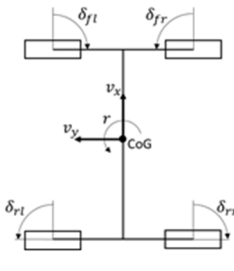
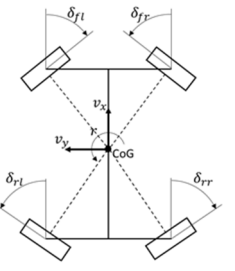
### 3.2.4 Overview of Driving Modes and Decision Making

Having established that the driving modes are tailored for certain driving situations the question is now in which hierarchical level of the control architecture a decision for the driving mode has to be made. First of all, it is necessary to define which circumstances have to be

given and which criteria have to be fulfilled in order to change a driving mode and this of course differs from mode to mode. But basically, a certain amount of information is necessary, e.g. the planned route, the global vehicle position or the dimensions and geometry of the environment. Furthermore, a certain driving characteristic can be defined as high comfort driving with a very low vehicle yaw rate or low turning circles in order to perform manoeuvres which need less space. When considering, for example, a parking scenario it is important to know the dimension of the parking spot. Depending on this information it can be evaluated if the vehicle can enter the parking spot with normal steering, special four-wheel steering or even only in sideways driving if the gap dimensions are very small. A similar situation can be found when the vehicle enters a narrow street or a dead end and has to turn around. If the street width doesn't allow to turn the vehicle around using conventional steering, turning on the spot needs to be applied. The use of either positive four-wheel steering or negative four-wheel steering can be decided upon the wanted driving characteristics or a demand for high maneuverability.

The examples above show that a decision for selecting a driving mode needs information, which is available at the navigation level. Therefore, it makes sense to make these decisions in the top level rather than e.g. in the motion planner. Since the focus of this thesis is on motion planner and trajectory follower, further details about the mode criteria and decision making won't be dealt with here. Table 5 shows an overview of the different driving modes and how they can be described using longitudinal velocity, lateral velocity and yaw rate in the vehicle coordinate system. In order to define a driving mode for the lower level controllers, the navigation level has to provide the three parameters and pass them on to the lower levels. From these three parameters, the mode can be identified in both motion planner and trajectory follower. With this information, the controllers can be adjusted accordingly with respect to the input and output constraints, which can vary from mode to mode.

**Table 5** Driving modes and vehicle velocity signals overview

Four-Wheel Steering Four-Wheel Drive		Sideways Driving	Turning on the Spot
Positive Steering	Negative Steering		
			
$-v_x \leq 0 \leq v_x$	$-v_x \leq 0 \leq v_x$	$v_x = 0$	$v_x = 0$
$-v_y \leq 0 \leq v_y$	$-v_y \leq 0 \leq v_y$	$-v_y \leq 0 \leq v_y$	$v_y = 0$
$r = 0$	$-r \leq 0 \leq r$	$r = 0$	$-r \leq 0 \leq r$

### 3.3 Motion Planning and Obstacle Avoidance

The fundamental task of the motion planner is commonly described as to establish a collision-free and feasible path or trajectory for the lower level controllers [1], [26], [22]. In that context, a path is a geometric description of the route within a global frame like e.g. a Cartesian coordinate system. Along this path, the vehicle velocity can be defined. A trajectory, on the other hand, is a geometric path to which a timing law is assigned, describing an evolution of the system over time [27]. An exemplary scenario for motion planning is that a path has to be planned within the vehicle environment in order to reach the goal point, considering constraints, avoiding collisions on the way and therefore ensure safe driving [1]. Regarding this scenario, the representation of the environment is important so that it can be used in the planning stage and that depends on the technologies used in the perception system and the further processing of the gained data. The approach in [22] for example is based on a visibility graph in which the shortest path is found using the A\* algorithm. The objects and lane markings in the vehicle environment are described by constraints, which can be used in the optimization problem of a Model Predictive Controller. Another way is discussed in [28] where a previously calculated trajectory is re-planned online in order to take changes in the vehicle environment and new situations into account.

The motion planner in this thesis receives a geometric path in global Cartesian coordinates and a vehicle longitudinal velocity from the navigation level. This approach was chosen in order to maintain a structure, which is similar to related projects at the research department of School of Automotive Studies from Tongji University. Therefore, a genuine path planning is not necessary for the motion planner stage since the task of computing a feasible and a basically collision-free path within the mapped environment is done one level above. Nevertheless, the term motion planner is chosen here since the obstacle avoidance takes place in this level and therefore a certain trajectory planning for the vehicle takes place. The motion planner can be described as a planner that takes the geometric path information from the navigation level and calculates fitting velocity and yaw rate parameters considering obstacle avoidance. The obstacle avoidance is done in a way that utilizes the ability of Model Predictive Control to handle constraints both on input and output variables, which are shown in a simulation example in Chapter 6.6. A simplified obstacle avoidance example derived from [29] is discussed here in order to show the basic idea behind the motion planner. When a vehicle is traveling on a multilane road, the left and right road boundaries represent the restrictions for lateral movement of the vehicle in order to not leave the road. These road boundaries can be described as constraints for the lateral displacement of the vehicle. Assuming that the vehicle has a perception system, which can detect obstacles on the lane the vehicle travels in and also measure the distance to the obstacle, a safety distance between ego vehicle and obstacle can be defined. This safe distance can again be formulated as a constraint and if this constraint is violated, actions have to be taken in order to not collide with the obstacle. A similar approach was chosen in [21]. These actions to be taken describe manoeuvres like braking or swerving by the obstacle. When performing a steering manoeuvre to pass by the obstacle, it is important to maintain a safe distance between vehicle and obstacle, not just because of collision avoidance as such but also due to the fact that only the vehicle center of gravity is taken into account on the course of the path and in order to represent the full dimension of the vehicle, the safety zones have to be defined accordingly. This can be realized by increasing the area occupied by the obstacle by a safety zone and then deriving



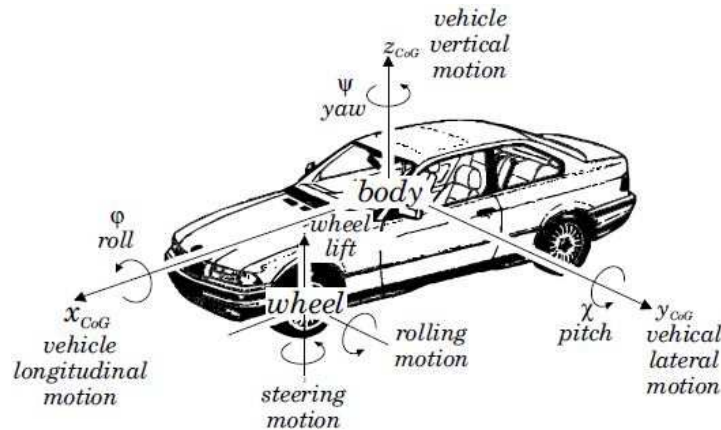
constraints from that. There are several different ways of approximating the shape of the objects on the road including a safety zone like e.g. a rectangular shape as described in [29], parabolas as used in [20] and collision cones [31], [30], just to name a few. Now the whole vehicle environment and the detected obstacle are defined by constraints in longitudinal and lateral vehicle direction, which makes it possible to perform the optimization algorithms in the Model Predictive Control strategy.

### **3.4 Trajectory Following**

As described in Chapter 3.3, the motion planner computes velocity and yaw rate inputs, which then have to be translated into commands for the vehicle. In the case of this thesis, these commands come from the trajectory following controller and are fed to the vehicle model used for simulations. The task of the trajectory follower is, therefore, to compute feasible control inputs by considering vehicle dynamics and actuator limitations. Thanks to the use of Model Predictive Control this can be achieved by defining appropriate constraints. Since the vehicle can perform several different driving modes as described in Chapters 3.2.1-3.2.3, the trajectory follower needs to facilitate a chassis control for each mode. Details about the realization of this control are shown in Chapter 4.2 and 4.3.

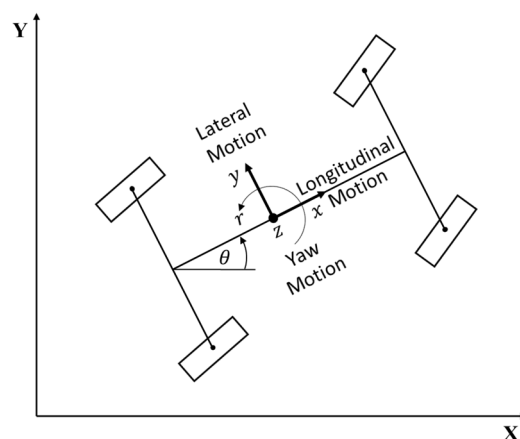
## 4 Prediction Models for Controller Design

In this chapter, the prediction models used for the MPC controllers in both motion planning and trajectory following level are introduced. They are all derived from the vehicle model used for simulation and show the characteristics of the 4WID-4WIS vehicle introduced in Chapter 1.3.



**Figure 4-1** The axis and movements in the vehicle coordinate system [32]

Figure 4-1 shows a vehicle and its coordinate system with the origin at the center of gravity. In general, a body moving in space has up to six degrees of freedom (DOF), which are the movement in  $x$ ,  $y$  and  $z$ -direction and a rotation around each of these three axis. Latter is described as yawing for rotation about the  $z$ -axis, rolling about the  $x$ -axis and pitching about the  $y$ -axis. Throughout this thesis, only vehicle dynamics in the  $x$ - $y$  plane is taken into consideration, which means that all movements in the  $z$ -direction, rolling and pitching are neglected. Therefore, the remaining three degrees of freedom of the vehicle body are displacement in  $x$  and  $y$ -direction and the yaw motion. Considering not only the vehicle body but also the four wheels, several additional degrees of freedom have to be added to the system, whereas in this thesis only the wheel's rotation around its lateral axis is considered as a degree of freedom, adding four additional DOF to the system. The wheel's rotation around its  $z$ -axis is the steering movement and hence it's an input to the model won't be considered as an additional degree of freedom [32]. The result is a seven DOF model, which is the case for the vehicle model used in simulations and which is described in Chapter 6.1.



**Figure 4-2** Planar vehicle model

A planar vehicle model in global  $X$  and  $Y$  Cartesian coordinates like it is shown in Figure 4-2 is used for the investigation of lateral vehicle dynamics [5]. In the case of this thesis, vehicle models for both obstacle avoidance and different driving modes are needed and models considering lateral vehicle dynamics were found to be suitable for that. In case of obstacle avoidance, the lateral vehicle movement is relevant in order to drive by the object and in correspondence to that also the lower level models were built as lateral vehicle models. The longitudinal vehicle dynamics, on the other hand, are relevant for driving and braking the vehicle. This aspect comes mainly into place when talking about safe braking by preventing the wheels from locking or accelerating the vehicle with no wheel spin. Since these factors are not directly related to the realization of obstacle avoidance and driving modes, they will not be considered in this thesis.

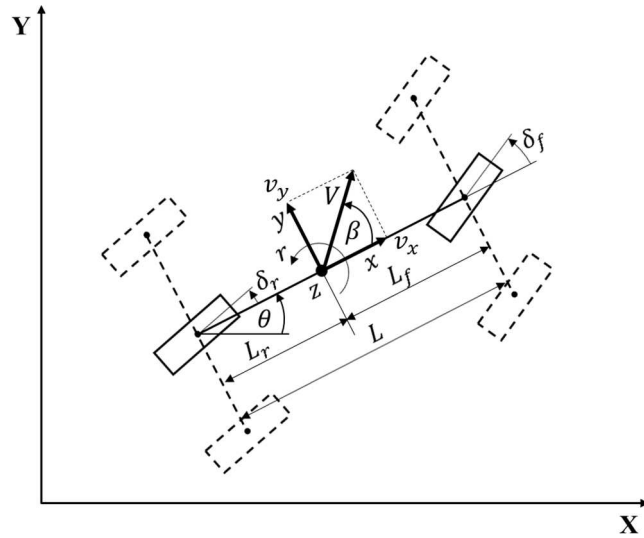
The controller design in MATLAB® requires a model formulation in discrete-time, Linear Time-Invariant (LTI) form, which means that it has to be established by either using a transfer function, state space or zero-pole-gain [17]. Therefore, all the mathematical models in this chapter are linearized and formulated in state space form.

## 4.1 Motion Planner

In the field of motion planning for vehicles using Model Predictive Control, several different kinds of prediction models are used considering the mathematical formulation. For example, [35] and [36] use a nonlinear kinematic model. Both [21] and [20] use nonlinear point mass models for path planning. The study conducted in [22] uses a linearized kinematic single-track model under small angle assumption and linear tire forces. Regarding the proposed control architecture in Chapter 3.1 and keeping the computational advantage of a simple vehicle model in mind, the decision was made to use a 2 DOF kinematic bicycle model for motion planning. As mentioned before, in order to design a Model Predictive Controller in MATLAB® using predefined functions an LTI model must be used. Therefore, the kinematic model is linearized under small angle assumptions and gets the longitudinal vehicle speed  $v_x$  as an input to the system.

### 4.1.1 Kinematic Bicycle Model for Motion Planning

In a kinematic vehicle model, the forces acting on the vehicle and influencing the vehicle motion are not taken into account. Instead, the vehicle motion is described by geometric relationships only. A simplification, which is made here is that both wheels of the front axle are combined in one single wheel positioned in the middle of the front axle with its center on the longitudinal vehicle axis. The same applies to the rear wheels. This representation is often called a single track vehicle model or bicycle model. Another assumption of this model, which is worth mentioning, is that the wheel sideslip angles are assumed to be zero, which is valid for low vehicle speeds and therefore small lateral tire forces (e.g. up to 5 m/s) [33]. Other more general modeling assumptions are e.g. that the mass of the vehicle is concentrated in the center of gravity, the load transfer between front and rear axle during driving is neglected and assumed to be distributed constantly [34].



**Figure 4-3** Kinematic bicycle model

Figure 4-3 shows a four-wheel steering bicycle model in global  $X$  and  $Y$  coordinates with the wheels of the axles combined to one single wheel each, derived from [34]. It is assumed, that the velocities act on the vehicle center of gravity (CoG), where  $V$  is the total vehicle velocity,  $\beta$  is the vehicle sideslip angle and  $\theta$  is the vehicle heading angle in relation to the global coordinate system. From this representation, the following equations can be derived in the global coordinate system, adapted from [5], [24].

$$\dot{X} = v_x \cos \theta - v_y \sin \theta \quad (4.1)$$

$$\dot{Y} = v_x \sin \theta + v_y \cos \theta \quad (4.2)$$

$$\dot{\theta} = r \quad (4.3)$$

With

$$v_x = V \cos \beta \quad (4.4)$$

$$v_y = V \sin \beta \quad (4.5)$$

and

$$\beta = \frac{v_y}{v_x} \quad (4.6)$$

Under small angle assumptions, the equations can be linearized.

$$\dot{X} = v_x - v_y \theta \quad (4.7)$$

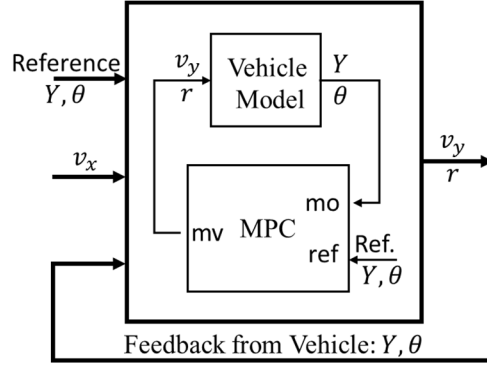
$$\dot{Y} = v_x \theta + v_y \quad (4.8)$$

For the state space representation of Equations (4.1)-(4.3) only the lateral and yaw dynamics are considered and the vehicle longitudinal speed  $v_x$  is seen as an input to the system, which reduces the complexity of modeling and therefore also of the controller. This results in a two DOF vehicle model. The inputs to the system are vehicle lateral velocity  $v_y$  and yaw rate  $r$  and the output are lateral vehicle position  $Y$  and heading angle  $\theta$  in the global coordinate system.

$$\begin{bmatrix} \dot{Y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & v_x \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} Y \\ \theta \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} v_y \\ r \end{bmatrix} \quad (4.9)$$

$$\begin{bmatrix} Y \\ \theta \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} Y \\ \theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} v_y \\ r \end{bmatrix} \quad (4.10)$$

Equations (4.9) and (4.10) show the state space of the kinematic bicycle model used as the prediction model in the MPC motion planner, referring to Equations (2.2) and (2.3).



**Figure 4-4** The inner and outer control loop of MPC Motion Planner

Figure 4-4 shows a schematic depiction of the kinematic bicycle model within the MPC structure of the motion planner. Here, the inner optimization loop uses the state space model described in (4.9) and (4.10) and gives the vehicle lateral speed  $v_y$  and the yaw rate  $r$  as manipulated variables (mv) to the trajectory follower one level below. The reference signal (ref), which should be tracked, is represented by the vehicle lateral position  $Y$  and the heading angle  $\theta$  in global coordinates and is compared in the MPC with the measured outputs (mo) from the kinematic bicycle model. In this thesis, it is assumed, that the vehicle position and heading angle in global coordinates can be measured, which is, of course, possible in the simulation using a mathematical model. It is unlikely, that the needed data is easily available in a real application since the global position and heading angle can't be measured directly. They need to be estimated based on traveled vehicle distance and sensor information.

## 4.2 Steering Controller

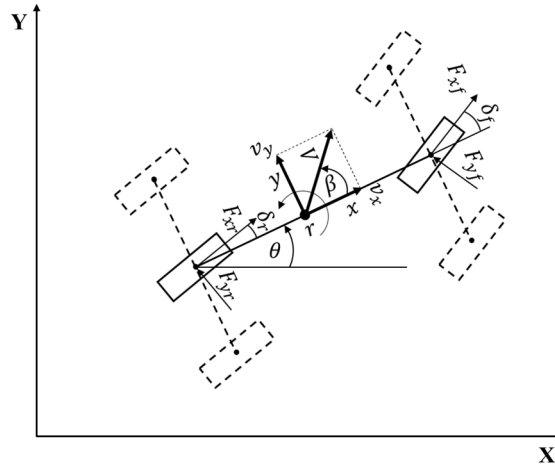
The motion planning level described above computes lateral vehicle speed  $v_y$  and yaw rate  $r$ , which then has to be translated into commands for the vehicle to follow. In the case of this thesis an advanced vehicle dynamic model described in Chapter 6.1 represents the controlled system and therefore, adequate control signals have to be computed in order to make the vehicle move along the planned trajectory. The advanced vehicle dynamic model is built in such a way that the steering angles of the four wheels represent the control inputs, next to the individual wheel torques. Due to this fact, it is obvious that a steering controller should be used for trajectory tracking and in order to take advantage of the special architecture of the 4WIS-4WID vehicle, it will be a four-wheel steering controller.

There are several studies and publications available, which deal with MPC steering controllers. In [37] and [16] a linear 2 DOF and four-wheel steering dynamic bicycle model is used and the steering angles are controlled using a geometric path namely global  $Y$  position and yaw rate as the reference to be tracked. Two other publications, research [28] and [6],

used a 3 DOF nonlinear bicycle model for two-wheel steering and they also use global vehicle position in  $Y$  direction and yaw rate as tracking reference. In [20] a nonlinear four-wheel vehicle model with two-wheel steering is used and the controller computes all four wheel-torque inputs additional to the front steering angle. A more simplified approach is chosen in [20] where a nonlinear kinematic bicycle model with two-wheel steering is used for path tracking. This thesis considers a four-wheel steering vehicle and the ability to control all steering angles is necessary for the four-wheel steering mode described in Chapter 3.2.1.

In order to keep the computational complexity of the controller low, a bicycle model is taken into consideration. With that, it is possible to compute both the front and rear steering angle but not all four steering angles at once. Given that both wheels on one axle turn into the same direction, as it is the case with the four-wheel steering mode, inner and outer wheel steering angle can be calculated from the single steering angle provided by the bicycle model. This applies to both positive and negative steering configurations. In addition to the question of computing the steering angles, the vehicle dynamics should be taken into account here in order to represent the actual vehicle in a sufficient way by still maintaining a certain simplicity, which helps to keep the computational complexity for the controller low. Concluding the aforementioned aspects and considering the signals, which are received from the motion planner, a 2 DOF dynamic bicycle model for vehicle lateral dynamics is chosen and as mentioned in Chapter 4.1 it is formulated as LTI model with the system input vehicle longitudinal speed  $v_x$ .

#### 4.2.1 Dynamic Bicycle Model for Steering Control



**Figure 4-5** 3DOF Dynamic Bicycle Model

Figure 4-5 shows the 3 DOF dynamic bicycle model in global  $X$  and  $Y$  coordinates and from that, the nonlinear Newton-Euler equations of motion can be derived. All equations in this chapter are derived and adapted from [24].

$$m(\dot{v}_x - v_y r) = F_{x_f} \cos \delta_f - F_{y_f} \sin \delta_f + F_{x_r} \cos \delta_r - F_{y_r} \sin \delta_r \quad (4.11)$$

$$m(\dot{v}_y + v_x r) = F_{y_f} \cos \delta_f + F_{x_f} \sin \delta_f + F_{y_r} \cos \delta_r + F_{x_r} \sin \delta_r \quad (4.12)$$

$$I_z \dot{r} = L_f F_{x_f} \sin \delta_f + L_f F_{y_f} \cos \delta_f - L_r F_{x_r} \sin \delta_r - L_r F_{y_r} \cos \delta_r \quad (4.13)$$

Equations (4.11)-(4.13) are established with respect to the vehicle center of gravity, whereas (4.11) describes the sum of the forces in the  $x$ -direction, (4.12) the sum of the forces in the  $y$ -direction and (4.13) the sum of the moments about the vehicle  $z$ -axis.  $I_z$  denotes the vehicle's moment of inertia about the  $z$ -axis and  $m$  is the total mass of the vehicle. These equations can be rewritten in the following way when assuming small steering angles.

$$m(\dot{v}_x - v_y r) = F_{x_f} + F_{x_r} \quad (4.14)$$

$$m(\dot{v}_y + v_x r) = F_{y_f} + F_{y_r} \quad (4.15)$$

$$I_z \dot{r} = L_f F_{y_f} - L_r F_{y_r} \quad (4.16)$$

For small sideslip angles,  $\alpha_f$  for the front wheel and  $\alpha_r$  for the rear wheel, the lateral tire force can be approximated by equations (4.17) and (4.18). Here, the tire cornering stiffnesses  $C_{\alpha_f}$  of the front wheel and  $C_{\alpha_r}$  of the rear wheel are a sum of the individual stiffnesses of each wheel, as can be seen in Equations (4.19) and (4.20).

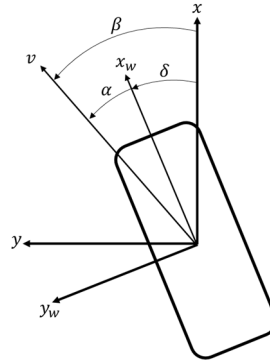
$$F_{y_f} = -C_{\alpha_f} \alpha_f \quad (4.17)$$

$$F_{y_r} = -C_{\alpha_r} \alpha_r \quad (4.18)$$

$$C_{\alpha_f} = C_{\alpha_f L} + C_{\alpha_f R} \quad (4.19)$$

$$C_{\alpha_r} = C_{\alpha_r L} + C_{\alpha_r R} \quad (4.20)$$

The sideslip angle  $\alpha_i$  of a wheel is composed of tire sideslip angle  $\beta_i$  and steering angle  $\delta_i$ . Figure 4-6 shows the angles at a wheel in the case of sideslip angle  $\alpha_i > 0$  and Equation (4.21) shows the relationship of the three angles.



**Figure 4-6** Angles of a cornering wheel

$$\alpha_i = \beta_i - \delta_i \quad (4.21)$$

The wheel sideslip angles for front and rear wheels can be expressed as:

$$\beta_f = \arctan\left(\frac{v_y + L_f r}{v_x}\right) \quad (4.22)$$

$$\beta_r = \arctan\left(\frac{v_y - L_r r}{v_x}\right) \quad (4.23)$$

Assuming small vehicle sideslip angle  $\beta$  and tire sideslip angle  $\beta_i$  the equation of the sideslip angle can be formulated as:

$$\alpha_f = \beta_f - \delta_f = \frac{v_y + L_f r}{v_x} - \delta_f \quad (4.24)$$

$$\alpha_r = \beta_r - \delta_r = \frac{v_y - L_r r}{v_x} - \delta_r \quad (4.25)$$

By inserting (4.17), (4.18), (4.24) and (4.25) one can rewrite (4.14), (4.15) and (4.16) into the following form:

$$\dot{v}_x = \frac{F_{x_f} + F_{x_r}}{m} + v_y r \quad (4.26)$$

$$\dot{v}_y = -\left(\frac{C_{ar} + C_{af}}{m v_x}\right) v_y + \left(\frac{C_{ar} L_r - C_{af} L_f}{m v_x} - v_x\right) r + \frac{C_{af}}{m} \delta_f + \frac{C_{ar}}{m} \delta_r \quad (4.27)$$

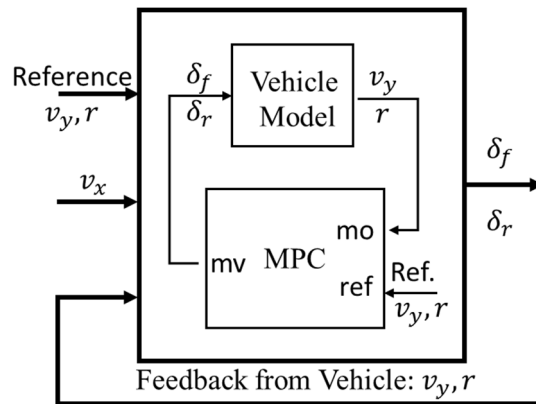
$$\dot{r} = \left(\frac{C_{ar} L_r - C_{af} L_f}{I_z v_x}\right) v_y - \left(\frac{C_{ar} L_r^2 + C_{af} L_f^2}{I_z v_x} - v_x\right) r + \frac{C_{af} L_f}{I_z} \delta_f - \frac{C_{ar} L_r}{I_z} \delta_r \quad (4.28)$$

For constant vehicle longitudinal speed  $v_x$ , Equations (4.26), (4.27) and (4.28) become linear and  $\dot{v}_x = 0$ . Therefore, Equations (4.27) and (4.28) become independent from (4.26). The state space representation of a four-wheel steering dynamic bicycle model for lateral dynamics can be seen in Equation (4.29) and (4.30) with constant vehicle longitudinal speed  $v_x$  as system input, referring to Equations (2.2) and (2.3).

$$\begin{bmatrix} \dot{v}_y \\ \dot{r} \end{bmatrix} = \begin{bmatrix} -\left(\frac{C_{ar} + C_{af}}{m v_x}\right) & \left(\frac{C_{ar} L_r - C_{af} L_f}{m v_x} - v_x\right) \\ \left(\frac{C_{ar} L_r - C_{af} L_f}{I_z v_x}\right) & \left(\frac{C_{ar} L_r^2 + C_{af} L_f^2}{I_z v_x} - v_x\right) \end{bmatrix} \begin{bmatrix} v_y \\ r \end{bmatrix} + \begin{bmatrix} \frac{C_{af}}{m} & \frac{C_{ar}}{m} \\ \frac{C_{af} L_f}{I_z} & \frac{C_{ar} L_r}{I_z} \end{bmatrix} \begin{bmatrix} \delta_f \\ \delta_r \end{bmatrix} \quad (4.29)$$

$$\begin{bmatrix} v_y \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_y \\ r \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_f \\ \delta_r \end{bmatrix} \quad (4.30)$$

This state space model is used as a prediction model in the Model Predictive Controller in a way, that the manipulated variables (mv) of the controller are adjusted so that the reference (ref) from the motion planner can be tracked. The feedback from the vehicle or vehicle model represents the measured output (mo) and is considered to be available even in real applications since it can be measured and calculated based on data from an inertial measurement unit (IMU). The schematic structure can be seen in Figure 4-7.



**Figure 4-7** The inner and outer control loop of MPC Steering Controller

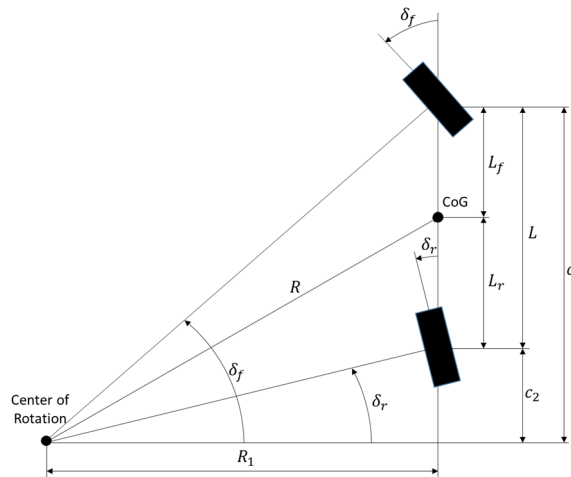


## 4.2.2 Ackermann Four-Wheel Steering

The steering controller presented in the previous chapter is based on a single track model, which only has two wheels and therefore only the front wheel and rear wheel steering angles can be computed. These steering angles are not fully suitable to be used as an input signal for a four-wheel vehicle model since the steering angle of the single wheel on an axle is not the same as the angle would be for two wheels on one axle. Furthermore, when cornering the steering angle of the inner wheel should not be the same as the angle of the outer wheel. If they would be the same, slip-free traveling would not be given and tire wear is increased. In order to avoid this effects, the kinematic steering, also known as Ackermann steering, can be used at low speeds [24]. In order to give appropriate steering angles to the controlled system, this chapter introduces the equations, which are needed to realize the Ackermann steering of four-wheel steering vehicles. The equations and figures in this chapter are derived and adapted from [24].

As shown in Chapter 3.2.1, the four-wheel steering mode can be performed with two different steering configurations, positive and negative steering. For each of these configurations, a different calculation has to be performed in order to realize Ackermann steering.

### Positive Steering



**Figure 4-8** Bicycle model with positive steering configuration

Figure 4-8 shows a positive steering bicycle model. The vehicle center of gravity (CoG) is traveling on a circle with radius  $R$  and the paths of both rear and front wheel lie on circles with the same center of rotation. The figure shows the geometric situation when performing a positive steering manoeuvre and from these triangles, the following equations can be derived:

$$\tan \delta_f = \frac{c_1}{R_1} \quad (4.31)$$

$$\tan \delta_r = \frac{c_2}{R_1} \quad (4.32)$$

Rearranging Equations (4.31) and (4.32) in a way that they both equal  $R_1$  and combining them results in Equation (4.33).

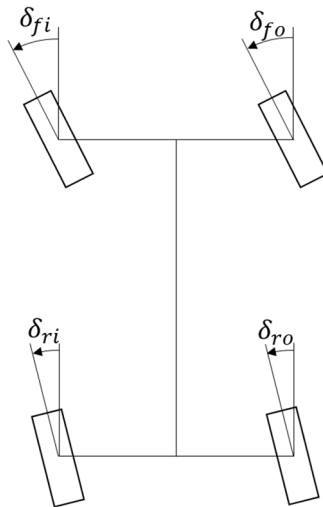
$$c_1 = c_2 \frac{\tan \delta_f}{\tan \delta_r} \quad (4.33)$$

With the geometric relationship in (4.34) using the wheel base  $L$ ,  $c_1$  can be expressed and all parameters in the equation are known. In the next step,  $c_1$  and  $c_2$  can be calculated (4.35).

$$c_2 = c_1 - L \quad (4.34)$$

$$c_1 = \frac{L \frac{\tan \delta_f}{\tan \delta_r}}{\frac{\tan \delta_f}{\tan \delta_r} - 1} \quad c_2 = L \left( \frac{\frac{\tan \delta_f}{\tan \delta_r}}{\frac{\tan \delta_f}{\tan \delta_r} - 1} - 1 \right) \quad (4.35)$$

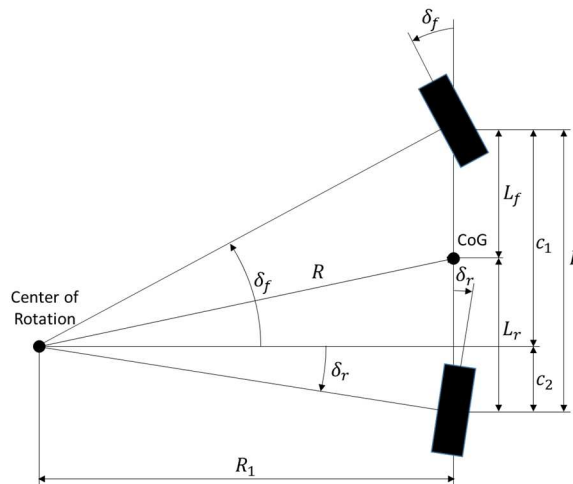
Knowing all these parameters, the kinematic steering angles for a four-wheel steering vehicle as shown in Figure 4-9 can be calculated using the set of equations in (4.36).  $B$  denotes the track width of the vehicle which is in case of the 4WID-4WIS vehicle the same for front and rear axle.



**Figure 4-9** Two-track vehicle with positive steering

$$\begin{aligned} \delta_{fi} &= \arctan \left( \frac{c_1}{R_1 - \frac{B}{2}} \right) & \delta_{fo} &= \arctan \left( \frac{c_1}{R_1 + \frac{B}{2}} \right) \\ \delta_{ri} &= \arctan \left( \frac{c_2}{R_1 - \frac{B}{2}} \right) & \delta_{ro} &= \arctan \left( \frac{c_2}{R_1 + \frac{B}{2}} \right) \end{aligned} \quad (4.36)$$

Negative Steering



**Figure 4-10** Bicycle model with negative steering configuration

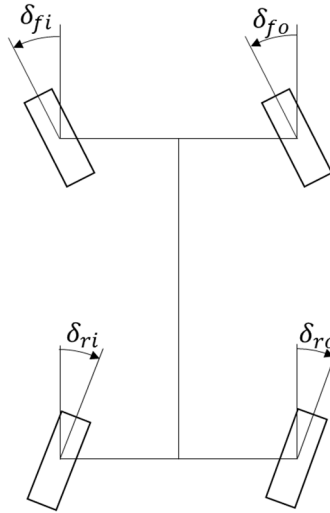
The negative steering configuration seen in Figure 4-10 shows a different geometric situation than the positive steering configuration discussed before because the rear wheel turns into the other direction than the front wheel. Nevertheless, the rules for kinematic steering still apply, which means that the vehicle center of gravity (CoG) is traveling on a circle with radius  $R$  and both rear and front wheel move along circles with their origin in the center of rotation. The triangles in Figure 4-10 are calculated the same way as for positive steering and can, therefore, be found in Equations (4.31), (4.32) and (4.33). The difference lies in the relationship between the wheel base  $L$ ,  $c_1$  and  $c_2$ , which is described in Equation (4.37).

$$c_2 = L - c_1 \quad (4.37)$$

With this relationship, the parameters  $c_1$  and  $c_2$  can be calculated for negative steering:

$$c_1 = \frac{L \frac{\tan \delta_f}{\tan \delta_r}}{1 + \frac{\tan \delta_f}{\tan \delta_r}} \quad c_2 = L \left( 1 - \frac{\frac{\tan \delta_f}{\tan \delta_r}}{\frac{\tan \delta_f}{\tan \delta_r} - 1} \right) \quad (4.38)$$

Now, the kinematic steering angles for a four-wheel vehicle with negative steering as shown in Figure 4-11 can be established by the set of equations in (4.39).



**Figure 4-11** Two-track vehicle with positive steering

$$\begin{aligned} \delta_{fi} &= \arctan\left(\frac{c_1}{R_1 - \frac{B}{2}}\right) & \delta_{fo} &= \arctan\left(\frac{c_1}{R_1 + \frac{B}{2}}\right) \\ \delta_{ri} &= -\arctan\left(\frac{-c_2}{R_1 - \frac{B}{2}}\right) & \delta_{ro} &= -\arctan\left(\frac{-c_2}{R_1 + \frac{B}{2}}\right) \end{aligned} \quad (4.39)$$

### 4.3 Torque Vectoring Controller

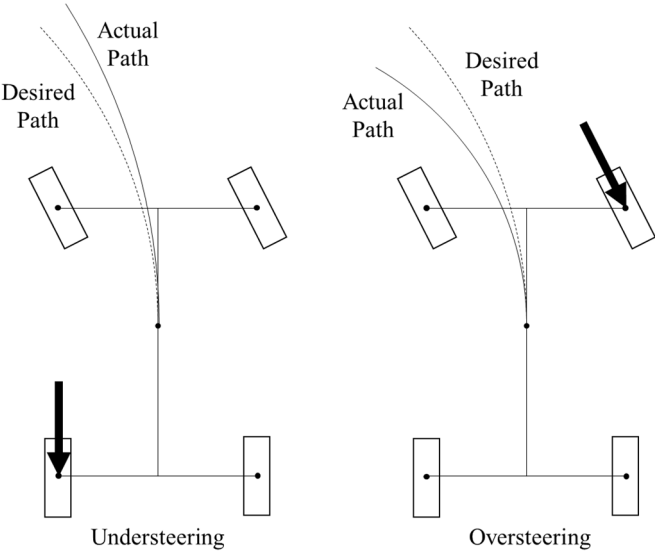
In Chapter 4.2 a steering controller is described, which can perform the trajectory following task by computing the vehicle steering angle inputs. It is mentioned, that the four-wheel steering mode can be realized, but the sideways driving mode and the turning on the spot mode can't. The reason for this is that in the last two modes the steering angles are predefined and mustn't change during the manoeuvre. Therefore, sideways driving and turning

on the spot modes can't be controlled with the aforementioned steering controller and only the individual wheel torques remain as controllable parameters. Tests with the vehicle prototype performing the sideways driving manoeuvre have shown, that it is crucial to adjust the steering angles properly to a right angle in order to drive in lateral vehicle direction without drifting off in longitudinal vehicle direction and, due to this, leaving the desired path. Such an error is e.g. caused by the mechanical clearance in the steering system. Another reason for a poor trajectory following performance in sideways driving is a certain difference in wheel speed between the individual wheels of the vehicle. This inequality can also cause the vehicle to not reach its desired yaw rate during the turning on the spot manoeuvre. Due to these effects on both sideways driving and turning on the spot, a wheel torque controller in torque vectoring fashion is introduced, which should help to counteract an undesired vehicle movement by influencing the vehicle yaw rate.

### 4.3.1 Torque Vectoring Principle

One of the characteristics of the 4WID-4WIS vehicle introduced in Chapter 1.3 is, that every wheel can be driven or braked individually and the wheel torque can be used as an input to the advanced vehicle dynamics model described in Chapter 6.1, which is used for the simulations. Due to that, the torque vectoring principle was studied in order to find a way of controlling the vehicle trajectory for sideways driving and turning on the spot.

The main objective of torque vectoring control is to stabilize the vehicle during driving and counteract unwanted steering tendencies like over- or understeering, which is achieved by applying braking torque to the wheels.



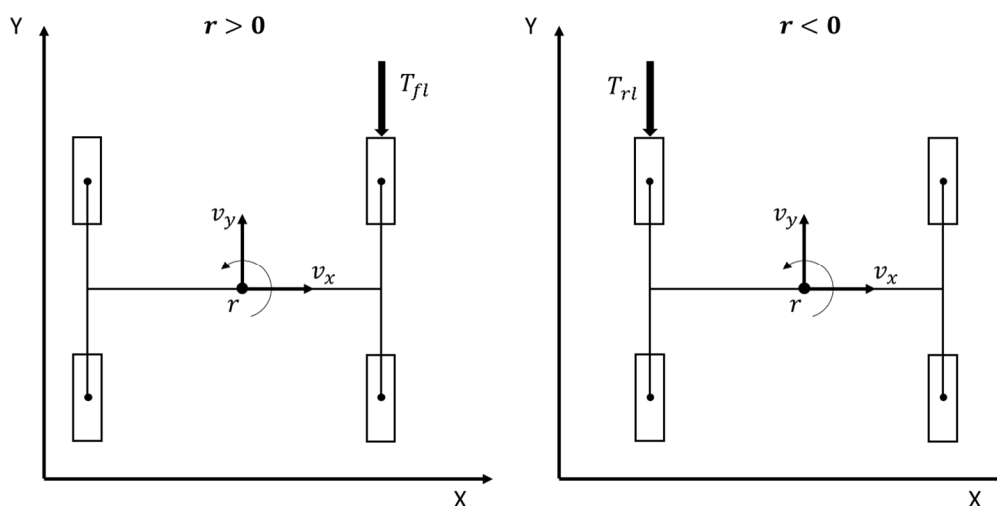
**Figure 4-12** Braking inputs to counteract steering tendency

Figure 4-12 shows the basic principle of torque vectoring for counteracting over- and understeering as it is shown in [15] and also described in [38]. A vehicle stability control would act against understeering by braking the inner left wheel and oversteering by braking the the outer front wheel.

There are several studies and publications available, which use Model Predictive Control for Torque Vectoring. In [38] an MPC yaw stability controller is designed based on a two-track vehicle model and simulated performing a standardized vehicle handling manoeuvre. The vehicle dynamic equations are simplified under small angle assumptions, small wheel slip and linear wheel forces. The publication [39] uses an extended bicycle model with linearized tire forces in the MPC lateral stability controller for path tracking. Instead of a conventional Model Predictive Control optimizing a quadratic program, they use matrix inversion to track a path. In [40] a linear Model Predictive Control strategy is used for the rear wheel torque vectoring in order to stabilize the vehicle at high lateral acceleration. The MPC controller computes the wheel slip ratios, which are then converted into torque commands via a sliding mode slip controller. In contrast to that, a linearized four-wheel vehicle model is used in [41] representing an independent four-wheel drive vehicle. The quadratic problem is formulated in a way, that the wheel torque distribution achieves a satisfying vehicle behavior.

The torque vectoring strategy in this thesis for the four-wheel steering mode is derived from [38] and is in line with the corrective torque inputs shown in Figure 4-12. When considering the front steering angle  $\delta_f$  computed by the steering controller to be greater than zero, the vehicle turns left. If in this case, the yaw rate is larger than the reference, the front right wheel will be braked and oversteering will be prevented. If in this scenario the yaw rate falls below the reference, the rear left wheel will be braked. Considering a right turn with  $\delta_f < 0$ , an exceeding yaw rate will be corrected by braking the front left wheel and brake torque will be applied to the rear right wheel if the yaw rate is below the reference.

The torque vectoring strategy for the sideways driving mode can be seen in Figure 4-13, where the braking inputs for driving in positive vehicle y-direction are displayed. As described at the beginning of this chapter, the vehicle drifts off its desired trajectory by badly adjusted steering angles or differences in wheel speeds. This drifting off causes a turning of the vehicle body and therefore a yaw rate. The control objective is to keep the vehicle yaw rate close to the reference value, which is zero in case of sideways driving.

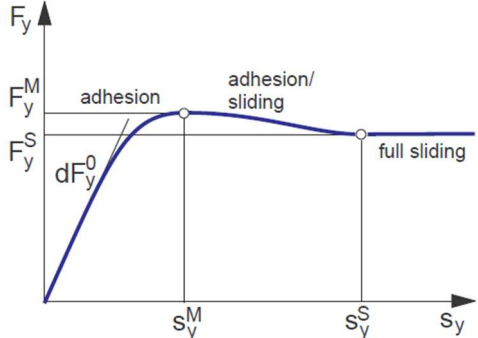


**Figure 4-13** Torque vectoring strategy for sideways driving in positive y-direction

As can be seen in Figure 4-13 in global  $X$  and  $Y$  coordinates, the front left wheel is braked if the yaw rate grows above zero, which means that it would turn counterclockwise

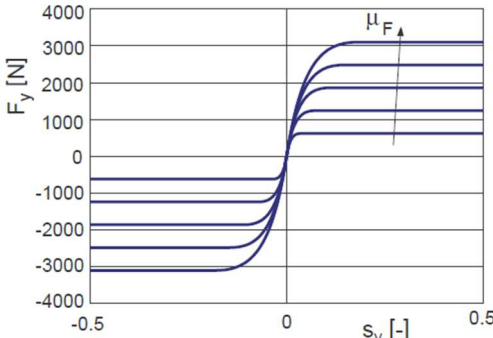
considering the left depiction. For a yaw rate less than zero, the rear left wheel has to be braked in order to counteract the clockwise rotation of the vehicle. When the vehicle is traveling in negative vehicle y-direction, the rear right wheel has to be braked in case of a yaw rate rising over zero. In the other case of a clockwise rotation, braking the front right wheel is necessary.

Under the above-stated assumption that the vehicle doesn't perform pure sideways driving due to certain errors, the motion of the vehicle occurs to a certain amount in lateral tire direction, which implies that lateral slip occurs. Basically lateral forces between ground and tires result in a lateral tire velocity and lateral slip. The lateral slip is the ratio between the velocities along the longitudinal and lateral tire axis [60].



**Figure 4-14** Lateral tire force over lateral slip [61]

The course of the lateral tire force  $F_y$  with respect to the lateral slip  $s_y$  can be seen in Figure 4-14, showing a linear and stable region with the maximum at  $s_y^M$  and  $F_y^M$ , followed by a mix of adhesion and sliding. With further rising lateral slip pure sliding occurs after reaching  $s_y^S$  at  $F_y^S$ . The initial inclination  $dF_y^0$  describes the cornering stiffness of the tire, which is one of the most important properties of the tire and relevant for vehicle handling and stability [61], [7].



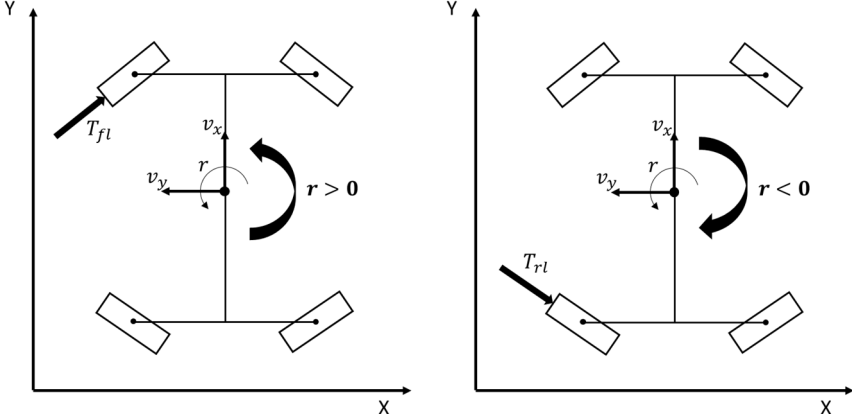
**Figure 4-15** Lateral force over lateral slip for different road friction coefficients [61]

The relationship between the lateral tire force  $F_y$  and the lateral slip  $s_y$  varies with the road friction coefficient  $\mu_F$ , as it can be seen in Figure 4-15. Another factor that influences the lateral slip and the ability of the tire to transmit forces is the vertical wheel load because the distribution of tire pressure over the wheel contact area changes with rising vertical load [61].

These characteristics of the tire in lateral direction influence the motion of the vehicle during sideways driving under the assumption of steering misalignment or wheel speed differences. The effects are also seen in the simulation discussed in Chapter 6.5.2.

As described at the beginning of this chapter, differences in wheel speeds of the four driven wheels can cause the vehicle to not reach the desired yaw rate during turning on the

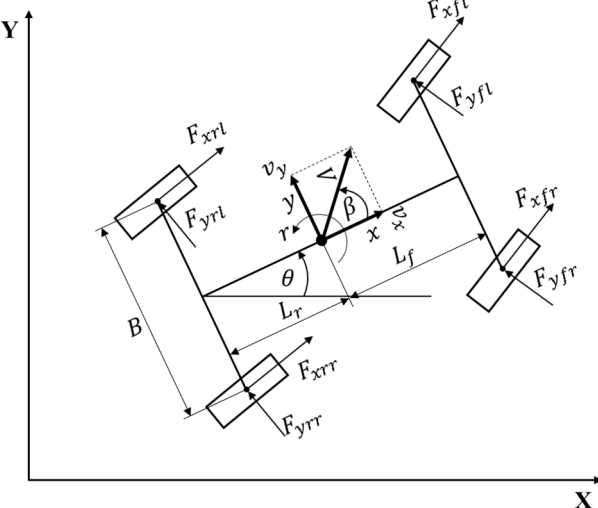
spot. Therefore, the torque vectoring strategy is applied, which aims at keeping the vehicle yaw rate close to the desired value, which is greater or smaller than zero depending on the wanted direction of rotation. In order to maintain a wanted yaw rate, a braking input is applied to one of the wheels, which can be chosen freely in contrast to sideways driving. This strategy is displayed schematically in global  $X$  and  $Y$  coordinates in Figure 4-16.



**Figure 4-16** Torque vectoring strategy for turning on the spot

4.3.2 Extended Dynamic Bicycle Model for Torque Vectoring

In order to understand the effect of individually applied braking force on the vehicle, it makes sense to have a look on the equations of a 3DOF four-wheel dynamic vehicle model.



**Figure 4-17** Dynamic two-track or extended bicycle model

Figure 4-17 shows the 3DOF dynamic two-track vehicle model in global  $X$  and  $Y$  coordinates. The depiction of the four steering angles was omitted here in order to maintain a clear structure. But it should be mentioned that in the following equations the front steering angle  $\delta_f$  and the rear steering angle  $\delta_r$ , computed by the steering controller will be used and not the steering angles calculated in Chapter 4.2. This is a simplification assuming that inner and outer wheel of one axle have the same steering angle, neglecting the Ackermann steering principle. By this assumption, the steering controller output can be used as input for the Torque

Vectoring Controller and the needed state space representation of the system can be formulated. Based on this model, the Newton-Euler Equations with the assumption of small steering angles can be established, similar to (4.15) and (4.16), [24].

$$m(\dot{v}_y + v_x r) = F_{yfl} + F_{yfr} + F_{yrl} + F_{yrr} \quad (4.40)$$

$$I_z \dot{r} = -F_{xfl} B_f + F_{yfl} L_f + F_{xfr} B_f + F_{yfr} L_f - F_{xrl} B_r - F_{yrl} B_r + F_{xrr} B_r - F_{yrr} L_r \quad (4.41)$$

In (4.41) the sum of moments acting on the planar vehicle is described, which equals the time derivative of the yaw rate multiplied by the vehicle's moment of inertia about the  $z$ -axis. In (4.16), a bicycle model was used and no forces in vehicle longitudinal direction are part of the equation, which is not the case in (4.41). Equation (4.40) on the other hand does not contain any forces in  $x$ -direction due to the small angle assumption. Therefore, it can be seen that considering these equations, additional brake forces to counteract unwanted steering behavior will have an influence on the vehicle yaw rate only.

The forces acting in vehicle longitudinal direction are unknown and therefore have to be approximated. Equation (4.42) shows such an approximation, which uses the tire longitudinal slip stiffness  $k_{xi}$  and the wheel slip  $\lambda$  [38]. The tire longitudinal slip stiffness is assumed to be about 50% larger than the tire cornering stiffness, which will be important for the simulations [7].

$$F_{xi} = k_{xi} \lambda_i \quad (4.42)$$

Applying the equations described in Chapter 4.2.1, namely (4.17), (4.18), (4.24) and (4.25), Equations (4.40) and (4.41) can be rearranged to the following form:

$$\dot{v}_y = -\left(\frac{2C_{\alpha r} + 2C_{\alpha f}}{m v_x}\right) v_y + \left(\frac{2C_{\alpha r} L_r - 2C_{\alpha f} L_f}{m v_x} - v_x\right) r + \frac{2C_{\alpha f} \delta_f}{m} + \frac{2C_{\alpha r} \delta_r}{m} \quad (4.43)$$

$$\begin{aligned} \dot{r} = & \left(\frac{2C_{\alpha r} L_r - 2C_{\alpha f} L_f}{I_z v_x}\right) v_y - \left(\frac{2C_{\alpha r} L_r^2 + 2C_{\alpha f} L_f^2}{I_z v_x} - v_x\right) r - \frac{k_{xf} B}{I_z} \lambda_{fl} + \frac{k_{xf} B}{I_z} \lambda_{fr} - \frac{k_{xr} B}{I_z} \lambda_{rl} \\ & + \frac{k_{xr} B}{I_z} \lambda_{rr} + \frac{2C_{\alpha f} L_f \delta_f}{I_z} - \frac{2C_{\alpha r} L_r \delta_r}{I_z} \end{aligned} \quad (4.44)$$

In Equations (4.43) and (4.44) the tire cornering stiffness is assumed to be the same for the front left tire and the front right tire, the same applies to the rear tires. The index of the wheel slip ratios  $\lambda_i$  refers to the wheel position on the vehicle, where  $i = fl, fr, rl, rr$ . The longitudinal slip stiffness is assumed to be the same for the tires of each axle.

For better readability, the constant terms in Equations (4.43) and (4.44) are substituted by letters.

$$\dot{v}_y = -A_1 v_y + A_2 r + B_1 + B_2 \quad (4.45)$$

$$\dot{r} = A_3 v_y - A_4 r - A_5 \lambda_{fl} + A_6 \lambda_{fr} - A_7 \lambda_{rl} + A_8 \lambda_{rr} + B_3 - B_4 \quad (4.46)$$

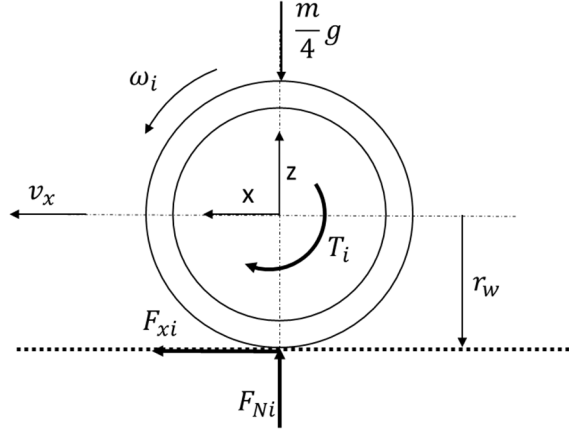
The wheel slip  $\lambda$  used in Equation (4.24) is different for braking and accelerating. The two situations are described in Equation (4.47) for braking and (4.48) for driving [42].

$$\lambda_B = \frac{v_x - r_w \omega_R}{v_x} \quad (4.47)$$

$$\lambda_D = \frac{r_w \omega_R - v_x}{r_w \omega_R} \quad (4.48)$$



For torque vectoring, the braking wheel slip is most relevant. Normally, Equations (4.47) and (4.48) would use the wheel longitudinal speed in the wheel coordinate system, which is here approximated as the vehicle longitudinal velocity  $v_x$  since small steering angles are assumed. Furthermore, the equations describing wheel slip normally take the dynamic wheel radius into account, which is here assumed to be the same as the radius of the wheel without load and equal for each wheel. In a next step, the forces and torques acting on the wheel are analyzed.



**Figure 4-18** Forces and moments acting on a moving wheel

Figure 4-18, derived from [42], shows the forces and torques acting on a moving wheel, where  $T_i$  is the braking torque,  $F_{Ni}$  is the tire normal load and  $(m/4)g$  describes a simplification in which the load on the wheels coming from the vehicle body are always distributed equally. Furthermore, rolling resistance is ignored. From this figure, Equation (4.49) for the angular velocity of the wheel can be derived [41].

$$I_w \dot{\omega}_i = -T_i - F_{xi} r_w \quad (4.49)$$

The derivative of the wheel slip can be described in a general form described in (4.50), derived from [43] and [38].

$$\dot{\lambda} = \frac{r_w \dot{\omega}_i v_x}{\max\{v_x^2, (r_w \omega_i)^2\}} \quad (4.50)$$

For braking, the following applies:

$$\dot{\lambda} = \frac{r_w \dot{\omega}_i}{v_x} \quad (4.51)$$

Therefore, Equation (4.49) can be rearranged to:

$$\dot{\lambda} = \left( \frac{r_w}{I_w v_x} \right) (-T_i - F_{xi} r_w) \quad (4.52)$$

And for driving, the wheel slip derivative is described as:

$$\dot{\lambda} = \frac{r_w \dot{\omega}_i}{v_x} (1 - \lambda)^2 \quad (4.53)$$

Which allows to write Equation (4.49) the following way:

$$\dot{\lambda} = \left( \frac{r_w}{I_w v_x} \right) (-T_i - F_{xi} r_w) (1 - \lambda)^2 \quad (4.54)$$

In this thesis, it is assumed that the wheel slip is kept within a stable region, which normally means that it is close to zero [38], [41], [43]. Therefore, the expression  $(1 - \lambda)^2$  can

be set to 1. With this assumption, both Equation (4.52) and (4.54) can be formulated the same way. Inserting Equation (4.42) in (4.52) and (4.54) the wheel slip of each wheel is established as it can be seen in the set of Equations (4.55).

$$\begin{aligned} \dot{\lambda}_{fl} &= -\left(\frac{r_w^2 k_{xf}}{I_w v_x}\right) \lambda_{fl} - \left(\frac{r_w}{I_w v_x}\right) T_{fl} & \dot{\lambda}_{fr} &= -\left(\frac{r_w^2 k_{xf}}{I_w v_x}\right) \lambda_{fr} - \left(\frac{r_w}{I_w v_x}\right) T_{fr} \\ \dot{\lambda}_{rl} &= -\left(\frac{r_w^2 k_{xr}}{I_w v_x}\right) \lambda_{rl} - \left(\frac{r_w}{I_w v_x}\right) T_{rl} & \dot{\lambda}_{rr} &= -\left(\frac{r_w^2 k_{xr}}{I_w v_x}\right) \lambda_{rr} - \left(\frac{r_w}{I_w v_x}\right) T_{rr} \end{aligned} \quad (4.55)$$

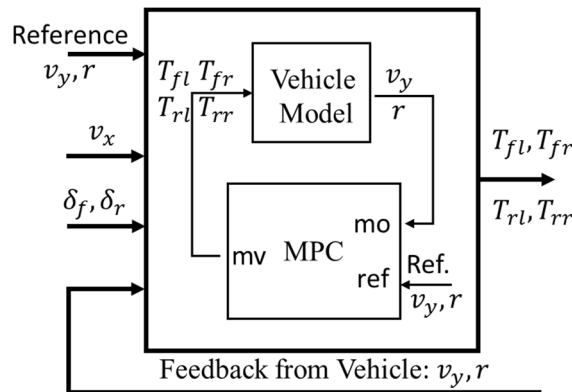
For better readability, the following form will be used for the state space formulation, with  $i = fl, fr, rl, rr$ .

$$\dot{\lambda}_i = -A_i \lambda_i - B_i T_i \quad (4.56)$$

Combining Equations (4.45), (4.56) and (4.55), the state space model for the torque vectoring controller can be built. Referring to Equations (2.2) and (2.3) the state space vectors and matrices are described below.

$$\begin{aligned} x &= \begin{bmatrix} v_y \\ r \\ \lambda_{fl} \\ \lambda_{fr} \\ \lambda_{rl} \\ \lambda_{rr} \end{bmatrix} & u &= \begin{bmatrix} n_1 \\ n_2 \\ T_{fl} \\ T_{fr} \\ T_{rl} \\ T_{rr} \end{bmatrix} \\ A &= \begin{bmatrix} A_1 & A_2 & 0 & 0 & 0 & 0 \\ A_3 & A_4 & A_5 & A_6 & A_7 & A_8 \\ 0 & 0 & A_{fl} & 0 & 0 & 0 \\ 0 & 0 & 0 & A_{fr} & 0 & 0 \\ 0 & 0 & 0 & 0 & A_{rl} & 0 \\ 0 & 0 & 0 & 0 & 0 & A_{rr} \end{bmatrix} & B &= \begin{bmatrix} B_1 & B_2 & 0 & 0 & 0 & 0 \\ B_3 & B_4 & 0 & 0 & 0 & 0 \\ 0 & 0 & B_{fl} & 0 & 0 & 0 \\ 0 & 0 & 0 & B_{fr} & 0 & 0 \\ 0 & 0 & 0 & 0 & B_{rl} & 0 \\ 0 & 0 & 0 & 0 & 0 & B_{rr} \end{bmatrix} \\ C &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} & D &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{aligned} \quad (4.57)$$

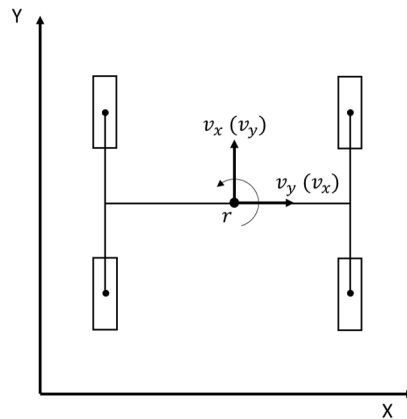
The input vector  $u$  in Equation Set (4.57) has two variables  $n_1$  and  $n_2$ , which are necessary to make the equation system work. They replace the steering angles, which are in this controller part of the state space matrices, as it is the case for the vehicle longitudinal velocity  $v_x$ . In order to have a low influence on the other variables,  $n_1$  and  $n_2$  are forced to the value of 1 by setting appropriate constraints in the Model Predictive Controller.



**Figure 4-19** The inner and outer control loop of MPC Torque Vectoring Controller

Figure 4-19 shows the implementation of the state space model in the Model Predictive Control for Torque Vectoring.

Almost all of the equations used to formulate the state space model have the vehicle longitudinal velocity  $v_x$  in the denominator. As described in Chapter 3.2.4 the sideways driving mode and the turning on the spot mode both require that  $v_x = 0$ , but that would cause a division by zero and has to be omitted. Therefore, two approaches were chosen.

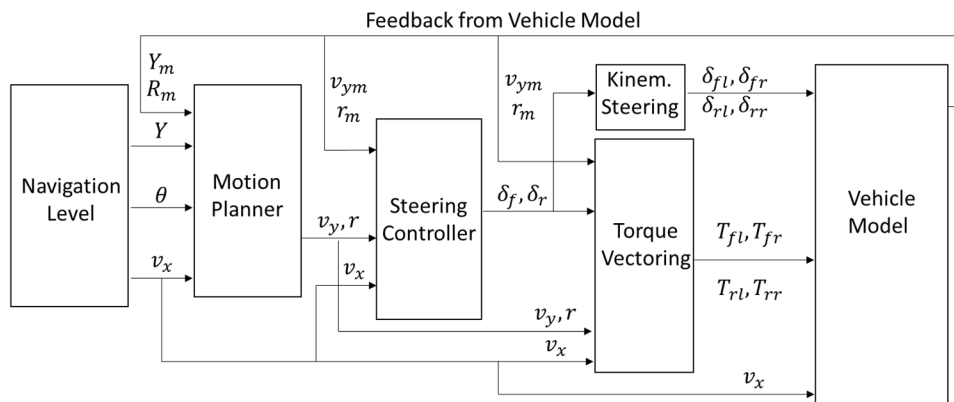


**Figure 4-20** The changed coordinate system for sideways driving

Figure 4-20 shows the vehicle in sideways driving mode in global  $X$  and  $Y$  coordinates, but with the changed vehicle coordinate system. By defining  $v_y$  as  $v_x$  and vice versa, the sideways driving vehicle appears for the controller like a vehicle with fixed steering angles, which is driving straight ahead. With that modification, the equations of the state space model still work and vehicle control in sideways driving mode is possible. For turning on the spot, on the other hand, such a reconfiguration can't be done, because both  $v_x$  and  $v_y$  are set to zero. A compromise is to set a small  $v_x$  value in order to make the system of equations work. By that one deliberately adds an error to the system, but since the error is caused on purpose, the effects could be studied and compensated in the planning stage at the navigation level. In Chapter 6.5.3 the simulations show the influence of  $v_x \neq 0$  when turning on the spot.

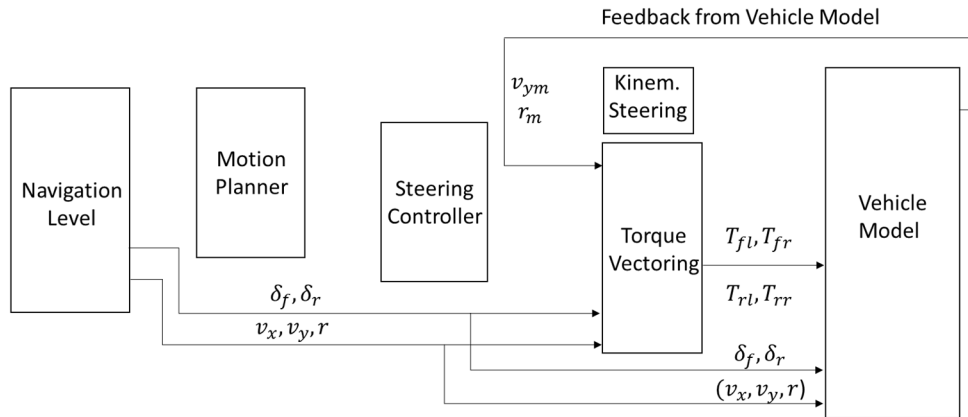
#### 4.4 Complete Control Structure

This chapter shows the complete control structure used in this thesis. It should help to visualize the signal flow and the hierarchy.



**Figure 4-21** Control structure for the four-wheel driving mode

Figure 4-21 shows the complete control structure for the four-wheel driving mode, which is also used for obstacle avoidance. In this configuration, all controllers are used. The vehicle longitudinal speed  $v_x$  is used as input to the vehicle model since this thesis doesn't use a slip controller, which takes care of the vehicle propulsion.



**Figure 4-22** Control structure for sideways driving and turning on the spot

Figure 4-22 shows the control structure used for the modes sideways driving and turning on the spot for which only the torque vectoring controller is used. Depending on the mode, the vehicle velocity and yaw rate signals are given by the navigation level and are in case input to the vehicle model. The steering angles are predefined by the mode settings.

This control structure is designed upon a number of simplifications and assumptions made with respect to the regarded field of application, described in Chapter 1.1. This includes that the vehicle is considered to be operated at low speeds and certain assumptions are based on that, e.g. the small angle assumption for linearizing the mathematical models. Therefore, a major limitation of this control structure is the ability to operate at higher vehicle speeds, namely above 50 km/h until which the vehicle was simulated in Chapter 6. The higher vehicle speed also implies a rise in vehicle dynamics, which can't be represented by the used linear vehicle models with a low number of degrees of freedom in a sufficient way. Due to this, the presented control structure is suitable for low vehicle dynamics only. The mathematical models used in this thesis asked for linearized tire forces, which is a major simplification since the tire forces are the biggest source of nonlinearity in a vehicle. Another simplification, which is a possible source of errors and limits the controller performance, lies in building the control models, whereas finding a balance between sufficiently representing the controlled system and maintaining simplicity for acceptable computation times is the main aspect. The simulation results in Chapter 6 show, that the linearized controllers can be used to control a nonlinear vehicle model within the considered operating range. Regarding sideways driving and turning on the spot, lateral vehicle speed and yaw rate were chosen in a low range since that reflects the situation in, for example, a sideways parking scenario. For applications using these modes, only low vehicle speeds and turning rates are suitable. For all of the previously described manoeuvres, it is crucial to bare the obstacle avoidance capabilities in mind which again ask for lower speeds in certain manoeuvres. For a real-world implementation of the presented control structure, a slip controller responsible for driving and braking the wheels in the stable slip zone has to be added. The error caused by setting  $v_x \neq 0$  during turning on the spot can be estimated by simulation and experiments and finally compensated during planning in the navigation level.

## 5 MPC Controller Design in MATLAB® and Simulink®

### 5.1 Controller Design in MATLAB® Command Line

The 4WID-4WIS electric vehicle introduced in Chapter 1.3 is modeled in Simulink® and therefore the simulations in this thesis are carried out using MATLAB® and Simulink®. The MPC controller design normally asks for a series of mathematical equations as described in Chapter 4. The Model Predictive Toolbox® from MATLAB® offers a simplified controller design based on a linear time-invariant (LTI) system model in the form of a transfer function, state space or zero-pole-gain [17].

At the beginning of controller design stands the state space formulation of the prediction model. This can be done by implementing the state space matrices  $A$ ,  $B$ ,  $C$  and  $D$ , which are described in Chapter 4 for the individual vehicle models and then turning them into a discrete state space by using the state space command as shown in Equation (5.1).

$$PredictionModel = ss(A, B, C, D) \quad (5.1)$$

Following this approach of controller design, it is recommended to check stability and controllability of the plant model. The stability can be checked by using the `isstable(PredictionModel)` command, which gives back  $0$  for an unstable plant and  $1$  for a stable plant [44]. For evaluating controllability, the `ctrb_matrix = ctrb(PredictionModel)` command can be used to calculate the controllability matrix, from which the rank can be computed by using `rank(ctrb_matrix)`. The plant is controllable if the rank of the matrix equals the number of states [45]. The following results are obtained for the state space models described in Chapter 4. Table 6 shows the results of the stability and controllability analysis of the state space models.

**Table 6** Results for stability and controllability checking

Prediction Model	Stability	Controllability
Motion Planner	1	2
Steering Controller	1	2
Torque Vectoring Controller	1	6

From Table 6 it can be concluded, that all three state space models which will be used for the controllers are stable and controllable. Knowing that the established prediction models in state-space representation are suitable for controller design, the `mpc` command can be used.

```
vx=2.5;                                %longitudinal speed
A=[0 vx; 0 0];
B=[1 0; 0 1];
C=[1 0; 0 1];
D=zeros(2,2);
PlantMP=ss(A,B,C,D);                   % state space model
```

```

PlantMP.InputName={'vy','r'};
PlantMP.StateName={'Y','Theta'};
PlantMP.OutputName=PlantMP.StateName;
PlantMP.InputGroup.MV=2;
PlantMP.OutputGroup.MO=2;

isstable(PlantMP); % check for stability
ctrb_matrix=ctrb(PlantMP);
rank(ctrb_matrix); % check for controllability

Ts1=0.017;
MPCobjMP=mpc(PlantMP,Ts1); % MPC Controller

```

The code lines shown above are from the MATLAB® command line and describe the first steps of the controller design process in case of the MPC motion planner. After the prediction model is transferred into state-space form, states, input and output of the system as well as the number of manipulated and output variables are defined. It should be mentioned here, that no disturbance or noise on both input and output signals is considered. After the previously described checking for stability and controllability, the sampling time can be defined and the MPC controller can be created. The code lines above are established by referring to MATLAB® online information and examples like [46] and [29] in order to have the right structure. After these steps, the MPC controller still has to be tuned and refined in order to work properly for the controlled system.

## 5.2 MPC Controller Tuning

In this chapter, several controller parameters are discussed and tuned. As a result, the controller is adjusted in order to work appropriately with the controlled system. In this thesis, the basic MPC controller tuning is done according to MATLAB® information and recommendations. The final adjustments are made in order to fulfill the criteria in controller *review* and the results turned out to be satisfactory, regarding the performance during simulations.

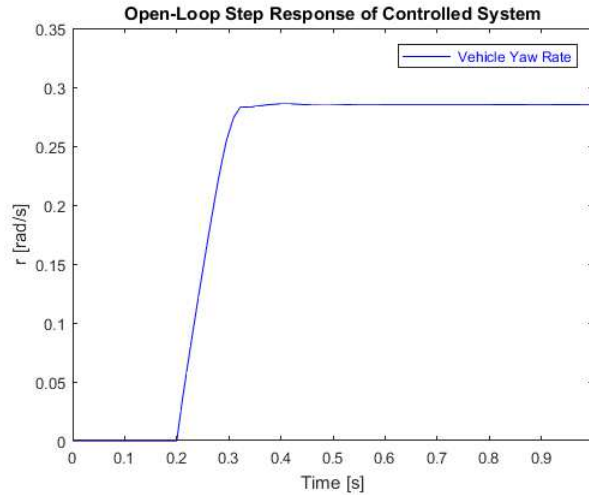
### Scale Factors

The specification of scale factors in Model Predictive Control is considered to be one of the basic adjustments in order to gain proper controller performance. It is necessary when variables both on input and output side have values with profoundly different magnitudes. The scale factors should represent the span of the regarded variable, which is the range between the upper and lower value. The MPC controller will then divide the signals by their scale factor, which results in dimensionless signals. Defining scale factors will make later tuning steps like facilitating cost function weights easier since the signal scale doesn't have to be considered anymore. Another advantage is that the influence of round-off errors will be lower during calculation [47].

### Sample Time

The length of the sampling time depends on the dynamics of the controlled system. For example, a system with slow dynamics like in the process industry will have a sampling time

considerably larger than one second, whereas the automotive field mostly deals with systems of high dynamics and therefore sample times below one second are common. In order to choose the sample time appropriately, the open-loop step response of the controlled system is analyzed [48]. In case of a vehicle, this is done by feeding a step steering input to the system, which are kept constant and recording the resulting yaw rate. The yaw rate will first rise, then oscillate around the steady-state yaw rate before it settles at a constant value. Considering the vehicle dynamics model described in Chapter 6.1 the step response looks as follows.



**Figure 5-1** Step response of the advanced vehicle dynamics model

Figure 5-1 shows the step response of the controlled system. By use of measuring tools, the rise time of the step response was identified  $T_{rise} = 0,085s$  and a settling time of  $T_{settle} = 0,335s$ . The sampling time should be between 10% and 20% of the rise time, as suggested in [48].

$$0,0085s \leq T_s \leq 0,017s \quad (5.2)$$

The advantage of a small sample time lies in the good rejection of disturbances on the system. The disadvantage is that a small sample time will increase the computational effort and therefore slow the system down or ask for bigger memory capacity. A compromise must be found between disturbance rejection and computational complexity [48]. Due to these reasons,  $T_s = 0,017s$  was chosen.

### Prediction Horizon

The length of the prediction horizon  $p$  gives the number of control intervals, which the MPC controller must take into consideration for the prediction of controller outputs [48], as it can be seen in Figure 2-1 of Chapter 2.1. It can also be compared to the look-ahead distance of a human driver. If the prediction horizon is chosen too small, changes in vehicle course or obstacles will be recognized too late. For a prediction horizon that is too large, sudden events like a pedestrian stepping onto the street can't be identified and therefore no reaction is possible [49]. Considering controller tuning, the prediction horizon can be calculated by using the sampling time  $T_s$  and the settling time  $T_{settle}$  of the steady-state system response [48], seen in Figure 5-1.

$$pT_s \approx T_{settle} \quad (5.3)$$

By using Equation (5.3) and a settling time of  $T_{settle} = 0,335s$  a prediction horizon of  $p \cong 20$  is defined. This value is chosen for the motion planner since it should utilize the full prediction horizon in order to handle obstacle avoidance. The values for steering and torque vectoring controller are chosen smaller, namely to a value of 10. This idea behind this decision is described in Chapter 3.1. In a later step, the review command will show that this choice of prediction horizons maintains internal controller stability.

### Control Horizon

A control horizon  $m$  describes the number of times new manipulated variables (mv) are calculated at the current control interval. Generally speaking,  $m$  should be between 1 and  $p$ , but [50] suggests to use at least 2 and maximum 4. The motion planner uses a control horizon of  $m = 4$ , steering and torque vectoring controller is set to  $m = 2$ . These decisions are in line with the prediction horizon lengths and the ideas presented in Chapter 3.1. Furthermore, a small control horizon helps to keep the computational complexity low because fewer variables have to be calculated at each control interval. It also helps to maintain internal controller stability [48].

### Constraints

In a Model Predictive Controller, the constraints are directly used in the optimization problem and therefore represent an important element of the whole MPC concept. The idea behind constraining input or output signals is that e.g. physical limitations of actuators in the controlled system are taken into account or that certain signals don't rise above a safe level. Besides representing the upper and lower bounds of a signal, constraints can also be applied on the rate-of-change of a signal. Constraints are described as inequalities and can be formulated hard or soft. Hard means, that the optimization has to keep the values within the defined limits. Soft constraining, on the other hand, allows the controller to violate the bounds in situations when it's necessary in order to fulfill the hard constraints. From that, one can conclude that hard constraints on both input and output variables might cause conflict in the optimization and no feasible solution is gained. Therefore, it is recommended that the manipulated variables are constrained hard whereas the output variable constraints should be kept soft. At this point, it should be mentioned that constraining the output variables should be omitted if possible and instead use a reference signal and adjust the cost function weights accordingly. Another remark on soft constraints is that if they are chosen too soft, an unacceptable violation of the constrained value could occur and could, therefore, cause unwanted reactions of the controlled system. If they are too hard, on the other hand, the controller might try to match the constraint even if its importance is low. Therefore, values of other and possibly more important system parameters might not be matched [51]. In this thesis, only the manipulated variables are constrained, hard for upper and lower limits and soft for the rate-of-change.

### Weights

The cost function weights of an MPC controller can help for example better track a reference signal by assigning weights between the inputs and outputs. By doing that, the controller characteristic can be influenced, like towards higher robustness if the input weights are higher than those on the output. A more aggressive and less robust behavior can be achieved by putting higher weights on the output signals than on the input [50]. During the



controller design in this thesis, weights are set on the output variables, which are considered as important for tracking, like e.g. the lateral vehicle position or velocity.

### Evaluation of Controller Tuning

MATLAB® offers a way to check the controller tuning quality with the command *review* [46]. This gives in particular information about the following aspects:

- MPC Object Creation
- QP Hessian Matrix Validity
- Closed-Loop Internal Stability
- Closed-Loop Nominal Stability
- Closed-Loop Steady-State Gains
- Hard MV Constraints
- Other Hard Constraints
- Soft Constraints
- Memory Size of MPC Data

MPC Object Creation checks, if the previously created controller fulfills all the requirements. The QP Hessian Matrix Validity shows, if the cost function parameters, prediction and control horizons are chosen in a way that the quadratic program (QP) has only one solution. Then the Hessian matrix of the quadratic program is positive-definite. This point includes the evaluation of weights on both manipulated and output variables, the horizons and the scale factors. Closed-Loop Internal Stability checks whether the prediction model itself is stable and if that stability is given in combination with the controller settings. The Closed-Loop Nominal Stability tests if the feedback connection between the controlled system and controller is stable in the unconstrained nominal operating point. Closed-Loop Steady-State Gains show if the controller manages to bring all output variables close to their target value at steady state without constraints. When the review command checks the system for Hard MV Constraints, it evaluates whether the hard constraints are violated in both cases of hard constraints on MV change and rate-of-change. When Other Hard Constraints are given, like on output variables, the controller might not be able to match them, because the hard constraints on manipulated variables are part of the QP problem and have priority. If Soft Constraints are defined in the controller, this review step checks if there is a proper balance between hard and soft constraints in the system. At the end of the MPC review, an estimate of the memory size for the online optimization of MPC is made, not considering the memory needed for source codes [52]. The review results for all three controllers are listed in Table 7. Three kinds of results can be distinguished, “Pass” if the test was successful, “Warning” if violations occurred and “Fail” if there is a fundamental problem or error in the MPC controller design.

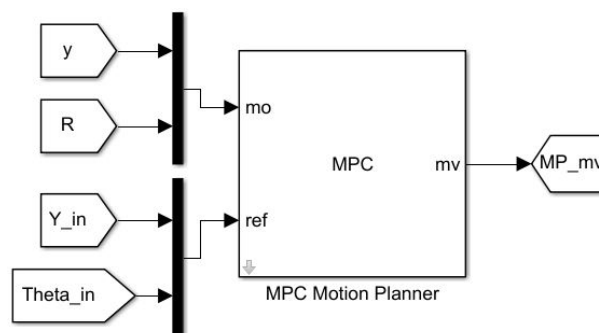
**Table 7** MPC review results for the three controllers

Performed Test	Motion Planner	Steering Controller	Torque Vectoring
MPC Object Creation	Pass	Pass	Pass
QP Hessian Matrix Validity	Pass	Pass	Pass
Closed-Loop Internal Stability	Pass	Pass	Pass
Closed-Loop Nominal Stability	Pass	Pass	Pass
Closed-Loop Steady-State Gains	Pass	Pass	Pass
Hard MV Constraints	Pass	Pass	Pass
Other Hard Constraints	Pass	Pass	Pass
Soft Constraints	Pass	Pass	Pass
Memory Size of MPC Data	10-40 kB	10-40 kB	20-60 kB

For memory size, the *review* command evaluates the case of standard MPC and MPC with online tuning. For each of those Single Precision (4 bytes) and Double Precision (8 bytes) is evaluated. In Table 7, only the lowest and highest values of all considered cases are displayed. It can be seen, that the torque vectoring controller needs more memory than the other two controllers since its state matrices have larger dimensions and therefore, the whole mathematical structure becomes more complex.

### 5.3 MPC Block in Simulink®

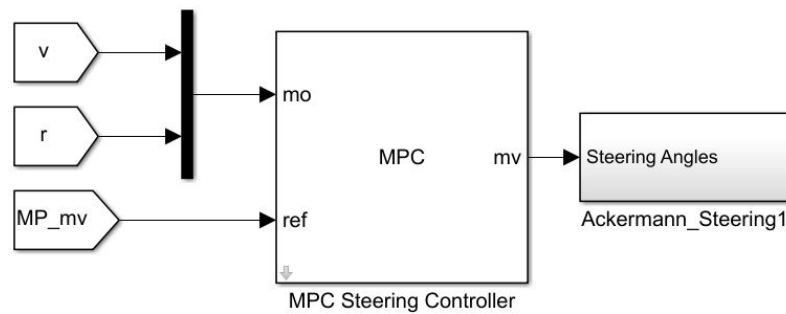
The Model Predictive Control Toolbox™ offers predefined blocks in Simulink®, which can be connected to the controlled system. In these blocks, the MPC controllers designed at the command line can be implemented and therefore used for simulations in Simulink®.



**Figure 5-2** Conventional MPC Block of the Motion Planner

In Figure 5-2 the MPC Motion Planner is depicted with the reference signals (ref) global  $Y$ -position and heading angle  $\theta$ , the measured output (mo) or state feedback from the vehicle

model, global vehicle  $y$ -position and vehicle heading angle  $R$ . The manipulated variables ( $mv$ ) are vehicle lateral speed  $v_y$  and yaw rate  $r$ .



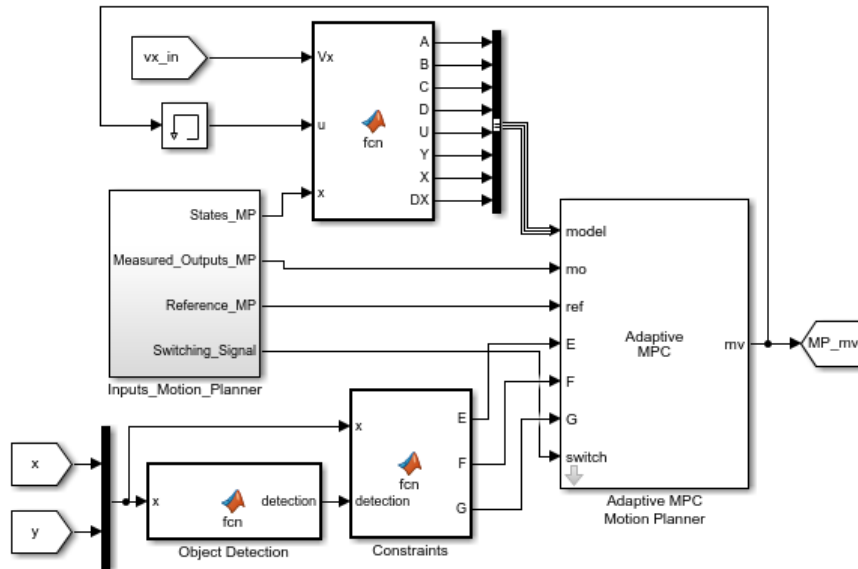
**Figure 5-3** Conventional MPC Block for the Steering Controller

Figure 5-3 shows the MPC Steering Controller. The reference signals ( $ref$ ) are the manipulated variables from the Motion Planner, vehicle lateral speed  $v_y$  and yaw rate  $r$ . The measured output ( $mo$ ) or state feedback from the vehicle model is therefore the measured vehicle lateral speed  $v$  and yaw rate  $r$ . The Steering Controller then computes the steering angles  $\delta_f$  and  $\delta_r$  as manipulated variables ( $mv$ ) and the Ackermann Steering block calculates the steering inputs for the four-wheel steering vehicle model.

The torque vectoring controller described in Chapter 4.3 can't be realized with a conventional MPC controller in Simulink<sup>®</sup>. The reason lies in the mathematical model inside the Torque Vectoring Controller which uses the vehicle longitudinal speed and the steering angles as constants in the state space formulation. The vehicle longitudinal speed can be considered constant in certain driving situations, but the steering angles must change in order to drive curves. Therefore, using a model with fixed steering angles isn't reasonable and an adaptive Model Predictive Controller has to be used, which is introduced in the next chapter.

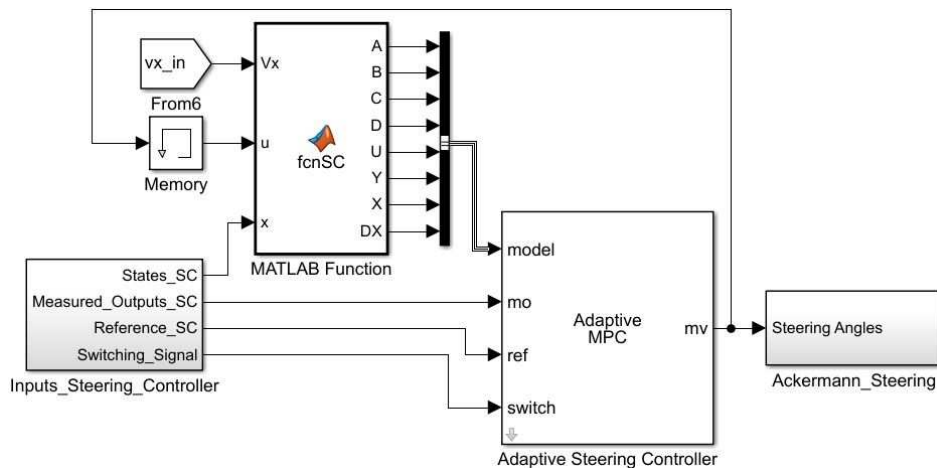
## 5.4 Adaptive MPC Block in Simulink<sup>®</sup>

The basic MPC controller designed in the command line uses an LTI plant model which is valid for a defined vehicle longitudinal speed  $v_x$  because the state space is formulated based on that. In the case of this thesis, a value of  $v_x = 2,5 \text{ m/s}$  is chosen, as can be seen in Chapter 5.1. Due to this, the controller will only properly work for this defined vehicle longitudinal speed and the performance will decrease for other velocities. The effect of changing longitudinal velocities is assessed in Chapter 6.2. In order to make the controller suitable for changing operating conditions like e.g.  $v_x$  or steering angles, Adaptive MPC can be used, which is especially suitable for handling nonlinear controlled systems over a wide range of operating conditions. The strategy behind Adaptive MPC is to use the same prediction model as for normal MPC but to update it together with the changed operating conditions at every control interval. The prediction model is kept constant over the prediction horizon. In order to use a MPC controller designed at the command line for Adaptive MPC, the continuous-time prediction model has to be converted into discrete-time. After updating the MPC controller with the discretized prediction model it can be used in the Adaptive MPC block in Simulink<sup>®</sup>, which is part of the Model Predictive Control Toolbox<sup>™</sup> [53], [54].



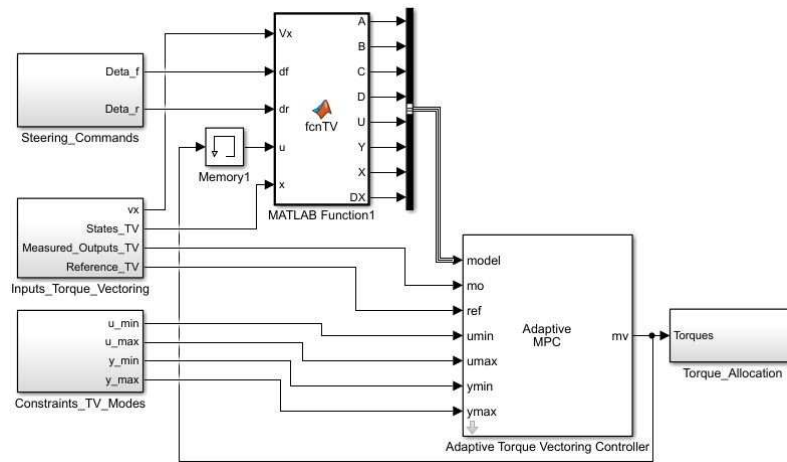
**Figure 5-4** Blocks of the Adaptive MPC Motion Planner

Figure 5-4 shows the Adaptive MPC Motion Planner implemented in Simulink®. It has a function, which updates the prediction model at every control interval, derived and adapted from [55]. This function uses the varying vehicle longitudinal speed  $v_x$ , the vehicle model states and the manipulated variables from the Adaptive MPC Controller as inputs to compute an updated prediction model for the controller. The Adaptive MPC block in Figure 5-4 has an input port called “switch”, which enables the user to switch off the optimization when the controller won’t be needed. This reduces the overall computational effort of the control structure. The internal states are still updated in order to enable optimization immediately after the controller gets switched on again [56]. Since the motion planner and the steering controller won’t be used in sideways driving and turning on the spot, they can be switched off during these manoeuvres. Three additional input ports called “E”, “F” and “G” are needed to enable obstacle avoidance and represent mixed input/output constraints.



**Figure 5-5** Blocks of the Adaptive MPC Steering Controller

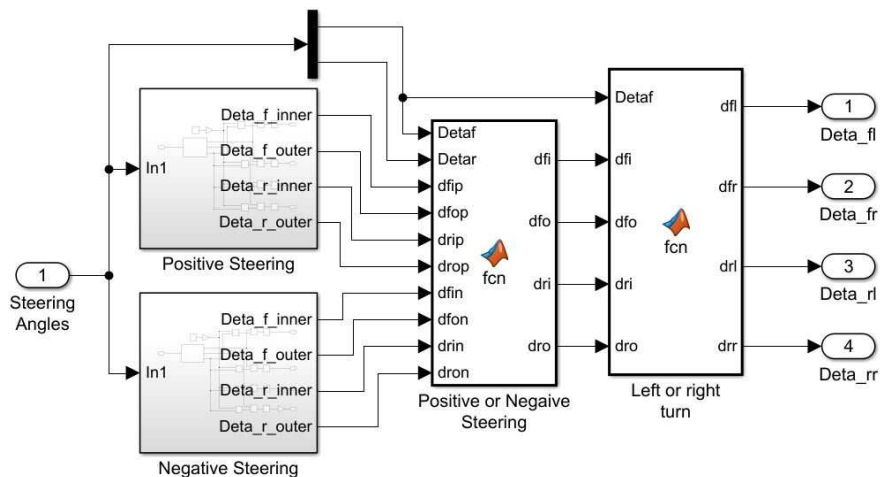
In Figure 5-5 the Adaptive Steering Controller can be seen, which has a similar structure as the Adaptive MPC Motion Planner described before. It also has a “switch” port to disable optimization when the controller is not needed. The manipulated variables are fed into the “Ackermann Steering” block. It can be seen in detail in Chapter 5.5.



**Figure 5-6** Blocks of the Adaptive MPC Torque Vectoring Controller

Figure 5-6 shows the Adaptive Torque Vectoring Controller, which uses the same model update strategy as the other two controllers but has additional inputs in form of the steering angles, which come directly from the Adaptive MPC Steering Controller. The Adaptive MPC block has in contrast to the two previously described controllers no “switch” port, because it can be used for all driving modes. But it has ports for the constraints on manipulated variables and output variables. This is necessary in order to enable the different torque vectoring strategies for each mode, as described in Chapter 4.3.1. The constraints will then only allow torque inputs for the wheel, which should be braked according to the torque vectoring strategy. The constraints on the output variables are all set to zero and, therefore, not considered in the controller as it was defined in the command line.

## 5.5 Kinematic Steering in Simulink®



**Figure 5-7** Calculation of the kinematic steering angles

The kinematic steering angle calculation from Chapter 4.2.2 is built in Simulink® and can be seen in Figure 5-7. A function detects if the steering controller computes steering angles in positive or negative steering. This is done by checking if both front and rear wheel steering angles have the same sign. After that, another function detects if the vehicle steers to the left or to the right by evaluating if the front steering angle is positive or negative. This is necessary in order to know, which wheels are the inner and outer wheels during cornering. Then the kinematic steering angles can be allocated to the wheels of the vehicle model.

## 6 Simulations

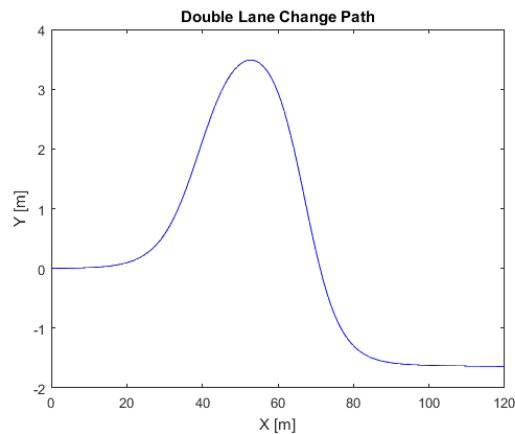
### 6.1 Advanced Vehicle Model used for Simulation

The 4WID-4WIS electric vehicle prototype from the School of Automotive Studies at Tongji University introduced in Chapter 1.3 was modeled by a student of the department in Simulink® for conducting simulations regarding vehicle dynamics and control [62]. The model considers the planar vehicle dynamics, similar to the ones discussed in Chapter 4 by taking into account 7DOF. These degrees of freedom are longitudinal, lateral and yaw movement of the vehicle body and the rotation of each wheel around its lateral axis. The steering angles of the four steerable wheels are considered as system inputs. This model was developed and tested at the department in order to understand the dynamic behavior of the vehicle considering the combination of four-wheel drive and four-wheel steering, for normal driving and the special driving modes described in Chapter 3.2. Besides design and simulation in Simulink®, a similar model was created in IPG Carmaker® and the test results were compared, confirming the suitability of the Simulink® model.

In order to represent the high nonlinearity of a real vehicle in modeling, the tire forces were described using a tire model called the “Magic Formula” from Pacejka [7]. Therefore, the linear MPC controllers in this thesis are used to control a nonlinear vehicle model and the results of this are shown in the next chapters.

### 6.2 Simulation Scenarios

A common scenario in vehicle simulation is the double lane change manoeuvre and it is widely used for assessing the performance of path tracking controllers [21], [6], [37], [16] and [28].



**Figure 6-1** Double lane change path

Figure 6-1 shows the path of the double lane change the vehicle has to follow. With Equations (6.1) and (6.2) [28] the path in global  $Y_{ref}$  direction and the heading angle  $\theta_{ref}$  can be calculated assuming a given longitudinal vehicle speed, which has to be kept constant in order to be used in a linear MPC.

$$Y_{ref}(X) = \arctan \left( d_{y1} \left( \frac{1}{\cosh z_1} \right)^2 \left( \frac{1,2}{d_{x1}} \right) - d_{y2} \left( \frac{1}{\cosh z_2} \right)^2 \left( \frac{1,2}{d_{x2}} \right) \right) \quad (6.1)$$

$$\theta_{ref}(X) = \frac{d_{y1}}{2} (1 + \tanh z_1) - \frac{d_{y2}}{2} (1 + \tanh z_2) \quad (6.2)$$

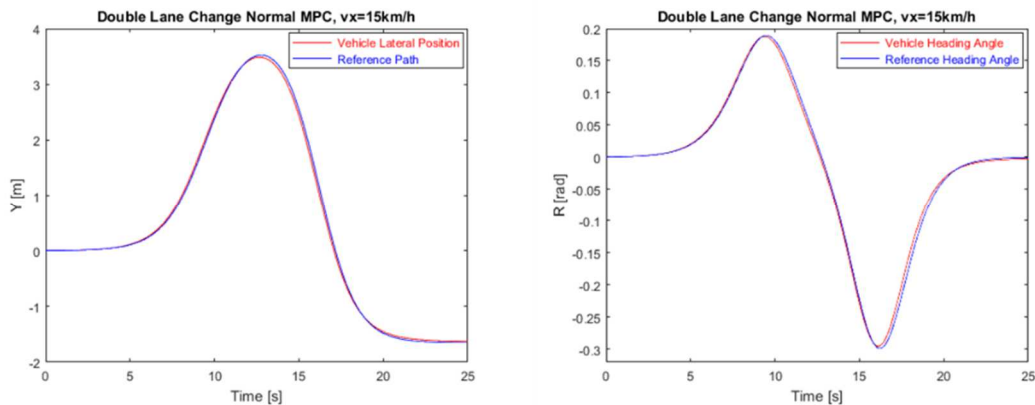
With the parameters  $z_1 = \frac{2,4}{25}(X - 27,19) - 1,2$ ,  $z_2 = \frac{2,4}{21,95}(X - 56,46) - 1,2$ ,  $d_{x1} = 25$ ,  $d_{x2} = 21,95$ ,  $d_{y1} = 4,05$  and  $d_{y2} = 5,7$ .

### 6.3 Simulation of MPC Steering Controller

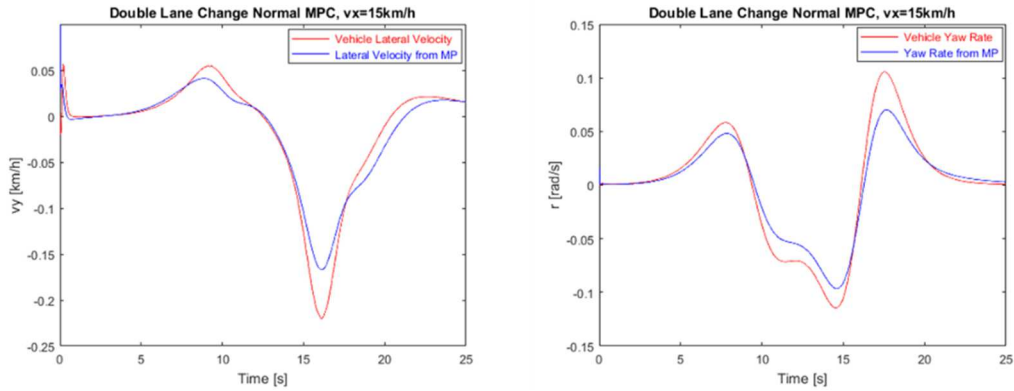
In this chapter, the performance of the steering controller considering trajectory following is investigated. For doing that, the global  $Y$  and  $\theta$  position of the vehicle model and the manipulated variables of the controller are checked and compared with the feedback values of the vehicle model. All simulations are performed at different vehicle longitudinal speeds of 15 km/h, 30 km/h and 50 km/h, in line with the field of application of an inner city vehicle. This is done in order to assess the capability of the controllers to handle varying vehicle speeds even though the prediction models are designed for  $v_x = 2,5 \text{ m/s}$  (or 9 km/h). All simulations in this chapter are done with the MPC Motion Planner and the MPC Steering Controller. The MPC Torque Vectoring Controller is not used here in order to assess the trajectory following capabilities of the steering controller alone.

#### 6.3.1 Double Lane Change with Normal MPC

The MPC Motion Planner and the MPC Steering Controller are used in these simulations and the effect of varying vehicle longitudinal speed  $v_x$  on the conventional or normal MPC controller is assessed.

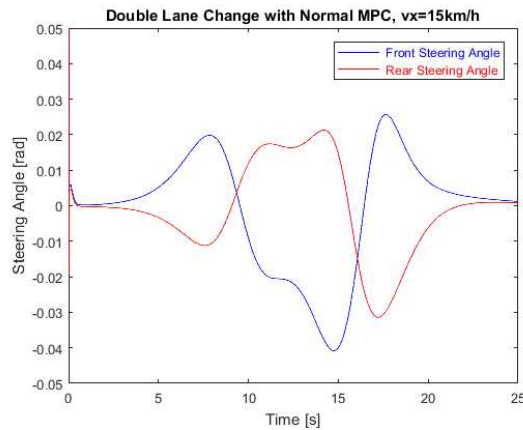


**Figure 6-2** Double Lane Change with normal MPC at 15 km/h



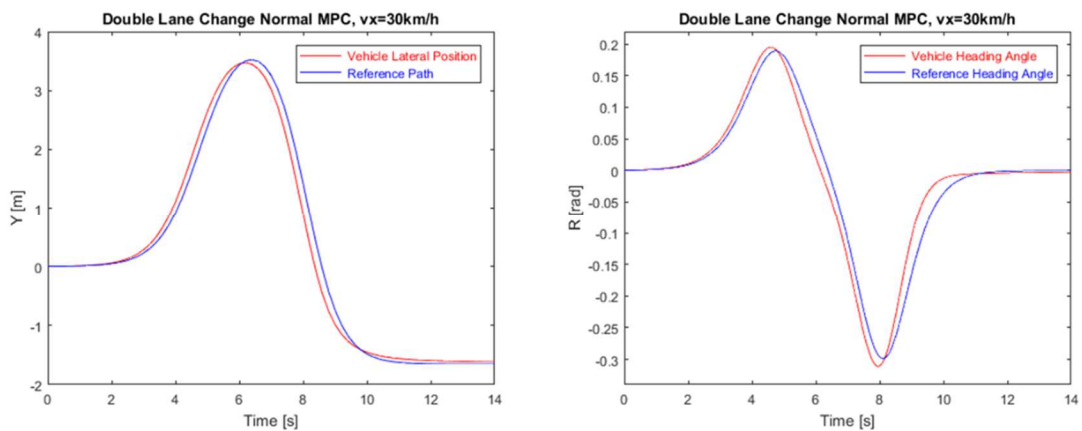
**Figure 6-3** Double Lane Change with normal MPC at 15 km/h

Figure 6-2 and Figure 6-3 show the double lane change manoeuvre at 15 km/h using conventional MPC Motion Planner and Steering Controller. The tracking performance seems acceptable but is not accurate since the vehicle speed  $v_x$  of 15 km/h already exceeds the speed at which the controller was designed.



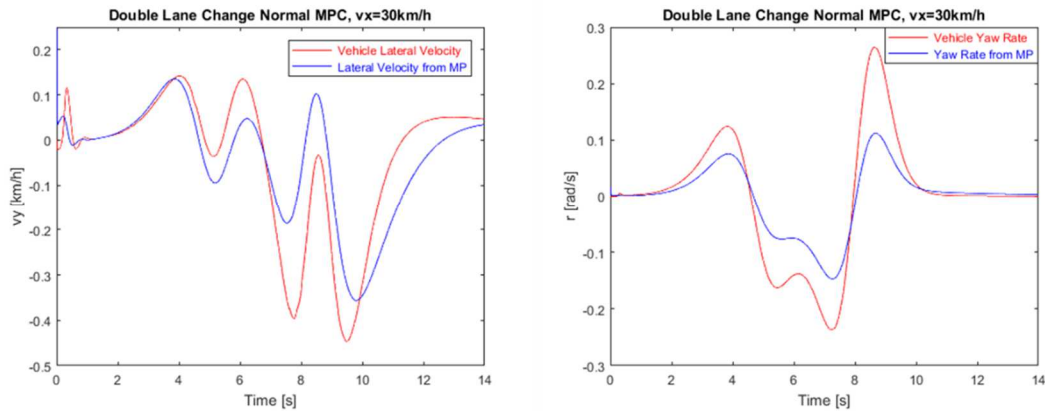
**Figure 6-4** Steering angles of the double Lane Change at 15 km/h

Figure 6-4 shows the change of steering angle for the double lane change manoeuvre at 15 km/h. It can be seen that the steering angles go in the opposite direction, indicating negative steering.



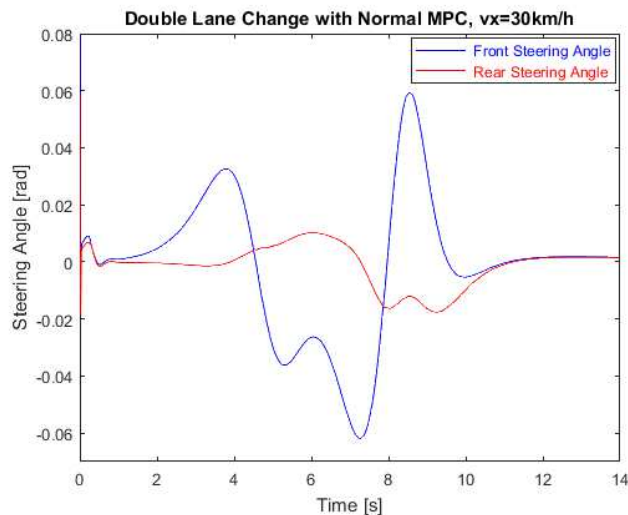
**Figure 6-5** Double Lane Change with normal MPC at 30 km/h





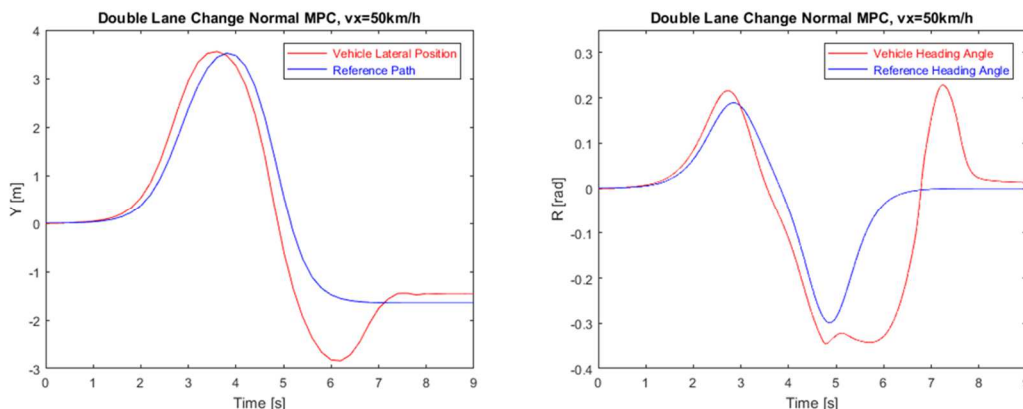
**Figure 6-6** Double Lane Change with normal MPC at 30 km/h

In Figure 6-5 and Figure 6-6 the simulation results for a double lane change at 30 km/h are presented. Compared to the manoeuvre at 15 km/h the tracking performance worsened and so do all the other signals. Both vehicle lateral speed  $v_y$  and yaw rate  $r$  show a big deviation between the reference values and the feedback signals from the vehicle model, indicating that the control structure operates outside its capabilities.

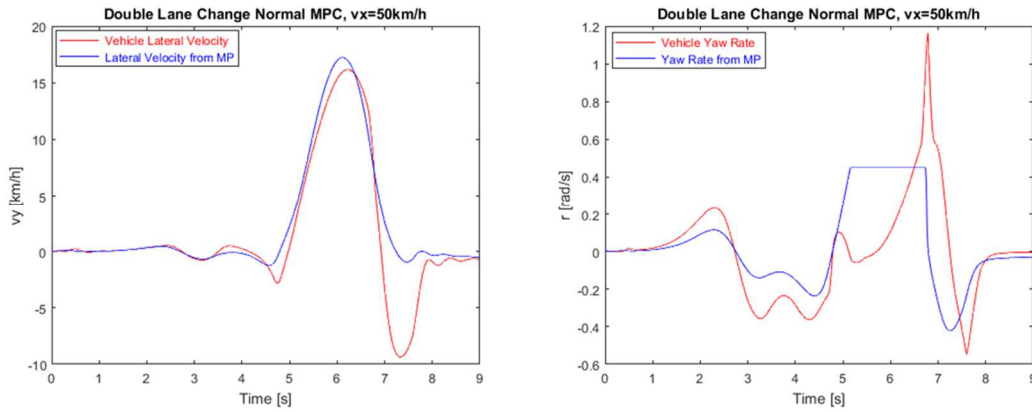


**Figure 6-7** Steering angles of the double Lane Change at 30 km/h

Figure 6-7 shows the steering angles of front and rear wheels computed by the MPC Steering Controller. The course of the signals is different than during the manoeuvre at 15 km/h but still feasible and within the physical boundaries

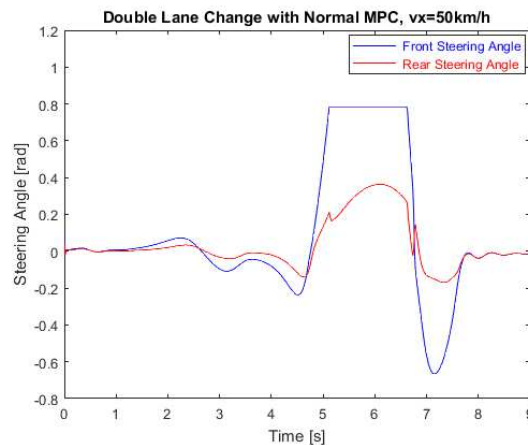


**Figure 6-8** Double Lane Change with normal MPC at 50 km/h



**Figure 6-9** Double Lane Change with normal MPC at 50 km/h

The graphs in Figure 6-8 and Figure 6-9 show the results of a double lane change manoeuvre at 50 km/h with the control structure composed of conventional MPC Motion Planner and Steering Controller. The tracking performance of the vehicle  $Y$  position and heading angle  $R$  is quite unsatisfying. The signal of lateral vehicle speed  $v_y$  is far higher than in any other simulation and probably too high for safe manoeuvring. In the bottom right graph, the vehicle yaw rate  $r$  is depicted, which in case of the computed signal from the Motion Planner reaches the upper constraint.

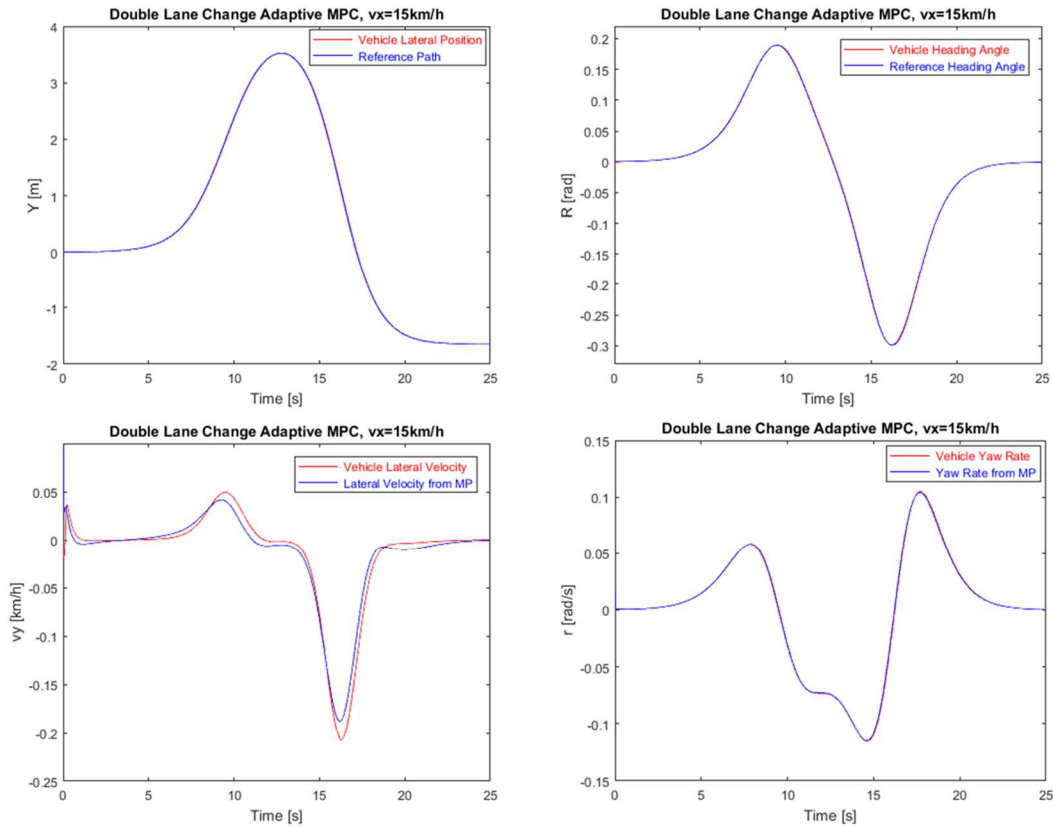


**Figure 6-10** Steering angles of the double Lane Change at 50 km/h

Figure 6-10 shows the steering angles of the double lane change at 50 km/h. The steering angle value of the front wheel reaches the upper constraint. The conclusion of simulations with conventional MPC controllers at various speeds is, that they are bad at or even unable to control the nonlinear vehicle model at speeds different from the velocity at which the controller was designed initially.

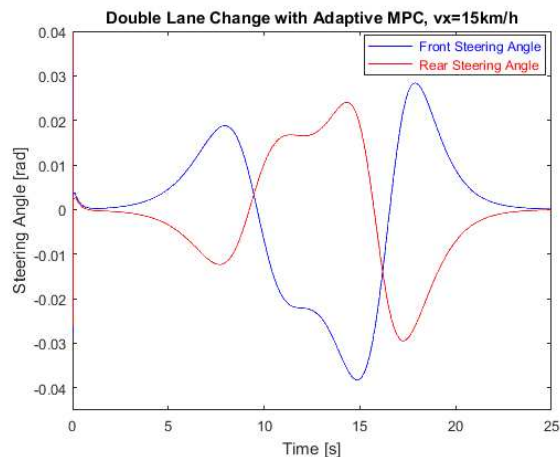
### 6.3.2 Double Lane Change with Adaptive MPC

In this chapter, the double lane change manoeuvre will be performed at different speeds in order to find out if the Adaptive MPC Controller is able to control the nonlinear plant at varying speeds.



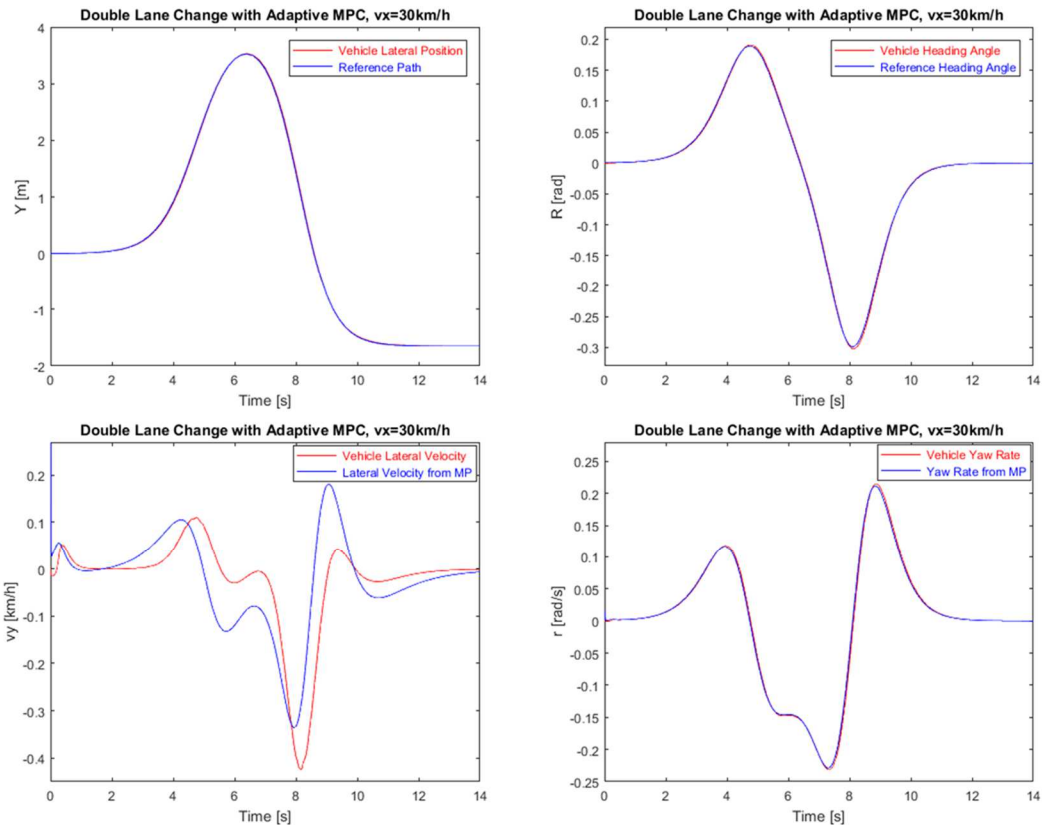
**Figure 6-11** Double Lane Change with Adaptive MPC at 15 km/h

The simulation results in Figure 6-11 show very good results for trajectory following during the double lane change manoeuvre at  $v_x = 15\text{km/h}$ . All signals are tracked with satisfying accuracy, even though the vehicle lateral speed  $v_y$  deviates slightly from its reference.



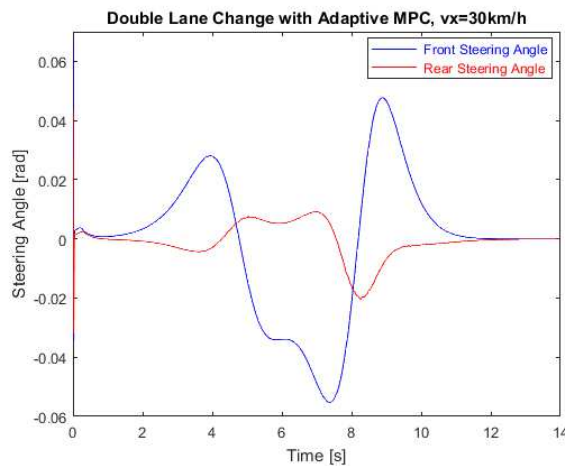
**Figure 6-12** Steering angles of the double Lane Change at 15 km/h

The front and rear wheel steering angles in Figure 6-12 show a smooth and feasible course and indicate negative steering since the wheels steer in opposite directions.



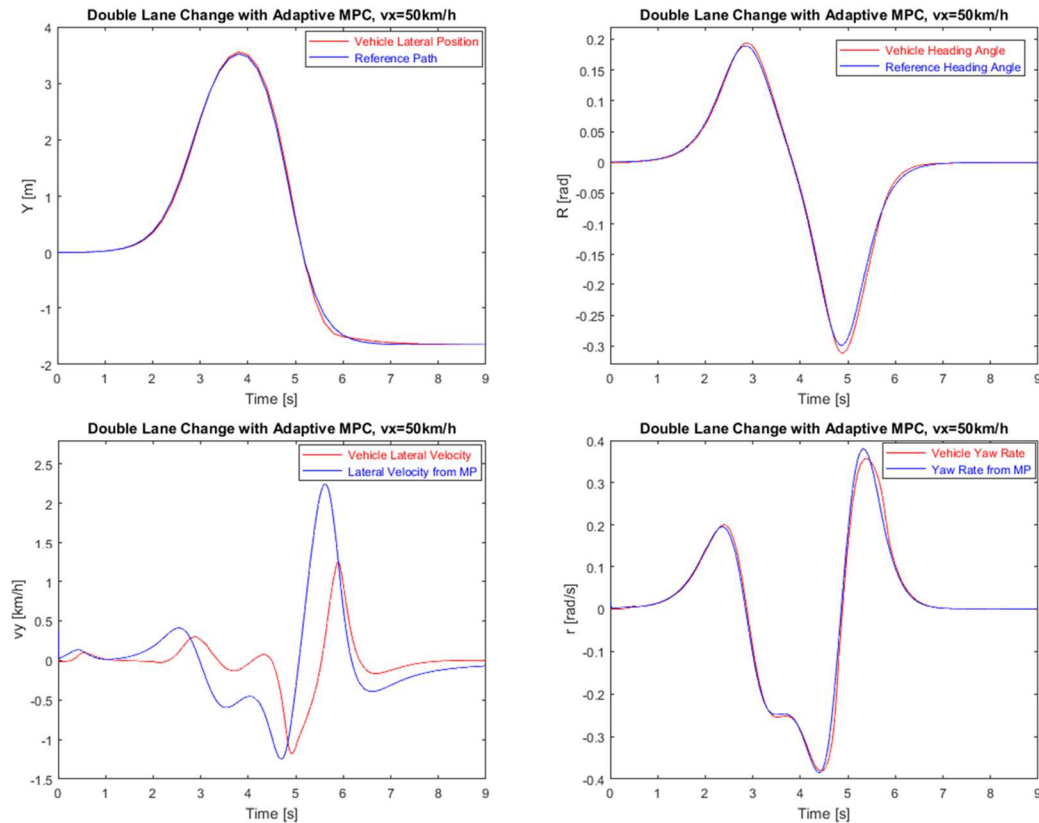
**Figure 6-13** Double Lane Change with Adaptive MPC at 30 km/h

Figure 6-13 shows the double lane change scenario performed a  $v_x = 30\text{km/h}$ . The tracking performance of lateral vehicle position  $Y$ , heading angle  $R$  and yaw rate  $r$  are very close to their references. Strong deviations can be seen between the vehicle lateral speed  $v_y$  and its reference.



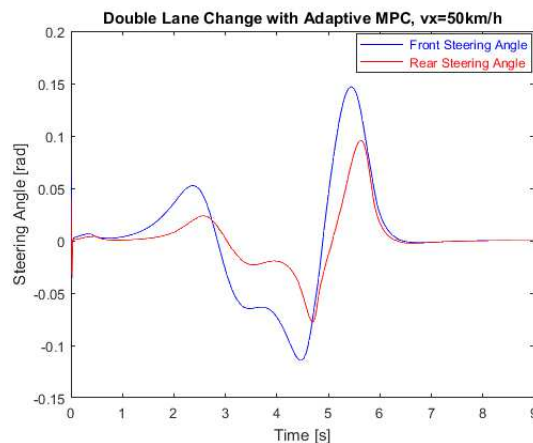
**Figure 6-14** Steering angles of the double Lane Change at 30 km/h

In Figure 6-14 the course of the vehicle steering angles computed by the MPC Steering Controller can be seen. Compared to the signals at 15 km/h, the ones at 30 km/h still show a basic tendency to negative steering and have a smooth curve, providing feasible control inputs for the vehicle model. The values of the front steering angle are bigger at 30 km/h than at 15 km/h.



**Figure 6-15** Double Lane Change with Adaptive MPC at 50 km/h

The simulation results of a double lane change manoeuvre at 50 km/h are shown in Figure 6-15. In terms of tracking the reference  $Y$  and  $\theta$ , the performance is still satisfying even though some deviations of the vehicle model feedback signals and the reference signals can be seen. The same can be said about the vehicle yaw rate  $r$ , only the vehicle lateral velocity  $v_y$  shows bad tracking performance, but the peak values are still far below those of the simulation with conventional MPC.



**Figure 6-16** Steering angles of the double Lane Change at 50 km/h

Figure 6-16 shows the computed steering angles at 50 km/h, which show in contrast to the simulations at a lower speed a tendency to positive steering. As mentioned in Chapter 3.2.1, positive steering can help to improve stabilization at higher speeds. The conclusion of these simulations is that the Adaptive MPC controllers are very capable of controlling the nonlinear vehicle model at various different speeds and show therefore a big advantage over

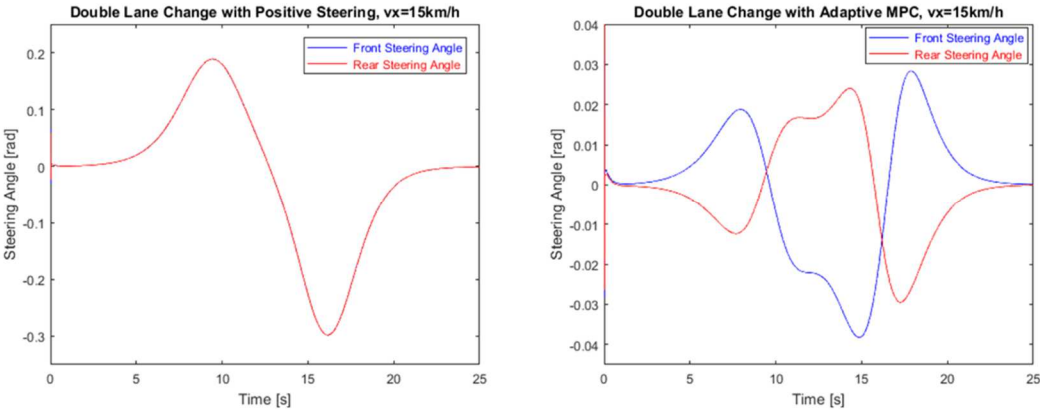
the common MPC controllers. Another conclusion from these simulations is that the tracking of vehicle lateral velocity  $v_y$  worsened with rising vehicle longitudinal speed and shows considerable deviations between the reference signal and the feedback signal. A reason for this behavior could lie in the way the MPC Motion Planner was designed since it only uses a kinematic prediction model, which is not able to represent a nonlinear vehicle model at high dynamics [36].

The Adaptive MPC steering controller shows a far better performance in trajectory following than the conventional MPC controller at various speeds. The reason for this is that the prediction model of the controller was established at a certain constant vehicle longitudinal speed. If this speed changes, the controller performance isn't optimal anymore and this is the case for conventional MPC. The Adaptive MPC updates the plant model at every time step with the changing vehicle longitudinal speed and is, therefore, able to represent the controlled system in a better way and the performance of the MPC control strategy improves significantly. It is, therefore, also capable of controlling the nonlinear vehicle model, even though the plant model of the MPC controller is linear.

### 6.3.3 Double Lane Change with Positive and Negative Steering

As described in Chapter 3.2.1, one of the characteristics of positive steering is that the angular movement about the vehicle vertical axis can be kept very small. Therefore, a double lane change simulation was performed with a reference vehicle heading angle  $\theta = 0$  and the yaw rate computed by the MPC Motion Planner was constrained to  $r = 0$ .

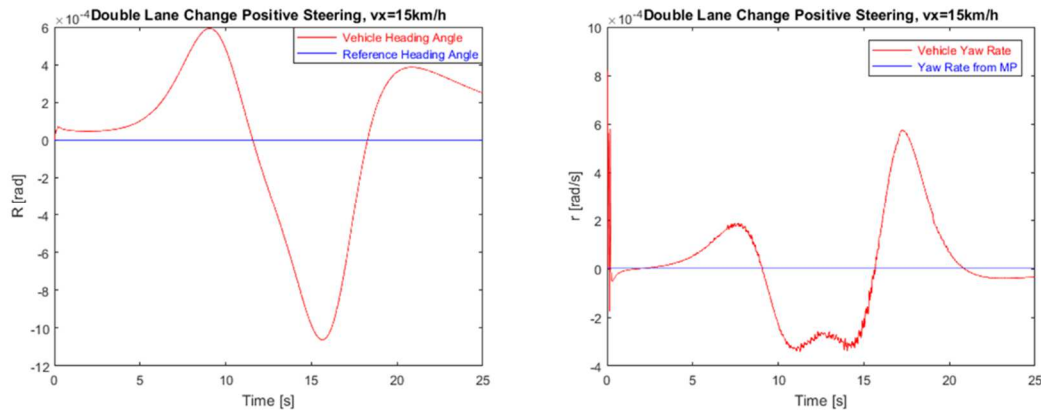
Negative steering configuration was achieved by performing the standard double lane change manoeuvre as described in Chapter 6.2 at 15 km/h during which the MPC Steering Controller computes steering angles in negative steering fashion.



**Figure 6-17** Front and rear steering angles for positive and negative steering

In Figure 6-17 the steering angles computed by the MPC Steering Controller can be seen for positive steering on the left and for negative steering on the right. The double lane change manoeuvre was carried out at 15 km/h. It can be seen that the steering angles of front and rear wheel are the same when performing a positive steering manoeuvre. For negative steering on

the other hand, the steering angles of the rear wheel have the opposite orientation than the front wheel.



**Figure 6-18** Heading angle and yaw rate during positive steering manoeuver

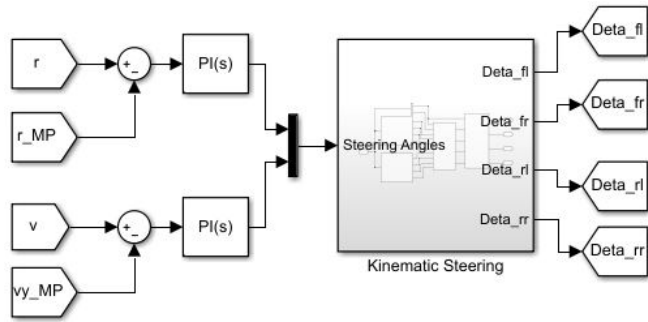
Figure 6-18 shows the results for the vehicle heading angle  $\theta$  and yaw rate  $r$  during the positive steering manoeuver. As described above, the reference values were set to zero. The feedback signals from the vehicle are not zero but still very small, as it can be seen in the graphs.

## 6.4 Simulation of a PID Steering Controller

After assessing the performance of the Adaptive MPC structure using Motion Planner and Steering Controller, a comparison is made using a PID controller for computing the steering angles. Because a PID controller can be implemented easily it is often the first choice for designing control structures and the easy implementation represents an advantage over the MPC controller. The results should show the difference between using an Adaptive MPC controller or a PID Controller for controlling a nonlinear plant. Similar to the previous simulations the double lane change manoeuver is performed at different longitudinal vehicle velocities.

### 6.4.1 Design of PID Steering Controller

In order to make a comparison with the MPC control structure, the Motion Planner is kept unchanged and is used for the simulations with the PID controller as well. The reason is that a suitable replacement with PID controllers was not found since the Motion Planner has to be capable of performing obstacle avoidance and that can be achieved with the MPC by updating the constraints. When looking to the MPC Steering Controller in Chapter 4.2.1, it uses the lateral vehicle speed  $v_y$  and the yaw rate  $r$  to compute both the front-wheel steering angle  $\delta_f$  and the rear-wheel steering angle  $\delta_r$ . Due to this, one PID controller computes  $\delta_f$  using the difference between reference vehicle lateral speed  $v_y$  and feedback lateral speed from the vehicle model. The second PID controller computes  $\delta_r$  with respect to the deviation of feedback and reference yaw rate. After conducting research in the field of four-wheel steering PID vehicle control a suitable concept was adapted from [57] where a PI controller is used. The following control structure is implemented in Simulink®.

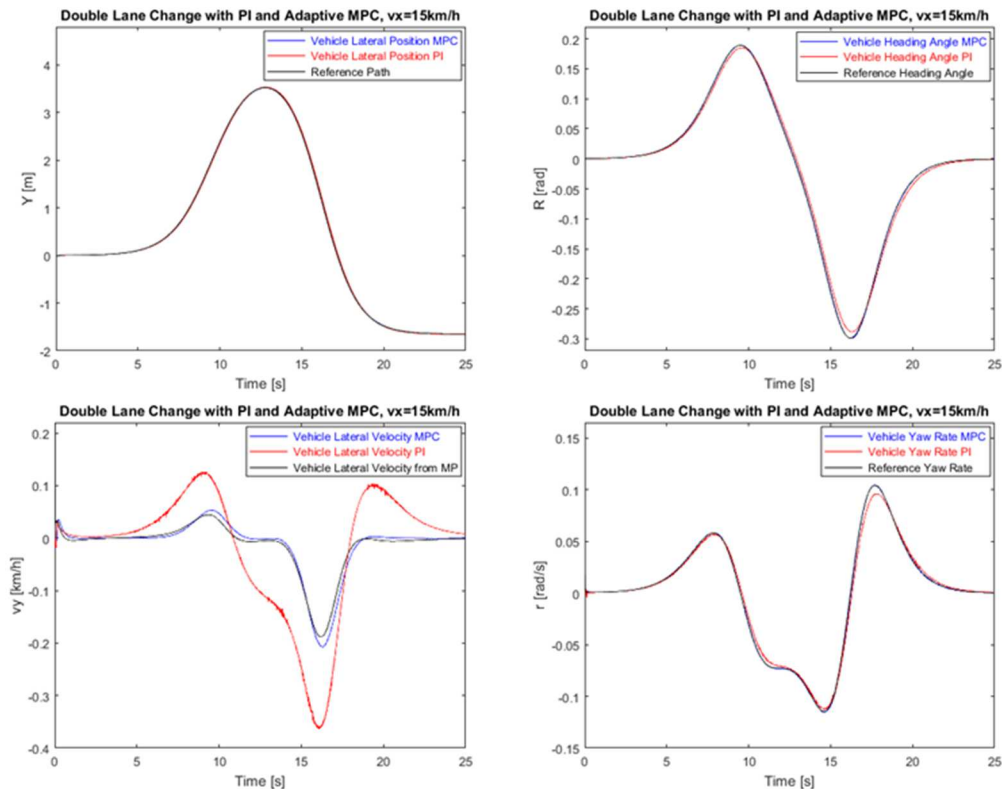


**Figure 6-19** The PI Controllers for four-wheel steering

Figure 6-19 shows the two PI controllers used for computing the steering angles, which receives the reference signals from the MPC Motion Planner. The PI controllers need to be tuned in order to work properly for the controlled system. Because the step response of the vehicle model was already used for MPC controller tuning in Chapter 5.2, the “T-Sum Rule” [58] was applied for setting the PI parameters and tuning them, which showed good results. In Simulink® it is possible to perform PID controller tuning, fully done by the software. This option was used during the PI controller implementation but showed no improvement to the “T-Sum Rule” and was, therefore, not used further on.

#### 6.4.2 Comparison with MPC Steering Controller and Conclusion

In order to compare the MPC Steering Controller and the PI Steering Controller, the double lane change manoeuvre was simulated at a vehicle longitudinal speed  $v_x$  of 15 km/h, 30 km/h and 50 km/h.



**Figure 6-20** Comparison of PI and MPC Controller for double lane change at 15 km/h



Figure 6-20 shows the simulation results of the double lane change at 15 km/h. The tracking performance of the PI Steering Controller can be compared with the one of the MPC Steering Controller, with only small deviations. The lateral vehicle speed  $v_y$  caused by the PI Steering Controller deviates strongly from the other signals.

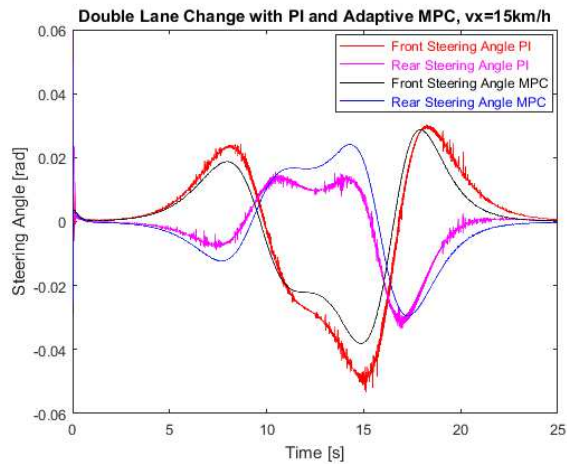


Figure 6-21 Comparison of computed steering angles at 15 km/h

Figure 6-21 shows the main difference in control quality between MPC Steering Controller and PI Steering Controller, which lies in the computed steering signal. The MPC controller computes smooth and feasible control inputs, which consider both constraints on steering angle range and turning rate. The results of the PI controller show strong fluctuation and won't provide smooth control inputs. This will be a problem if used for a real vehicle since the actuators won't be able to follow the computed steering commands.

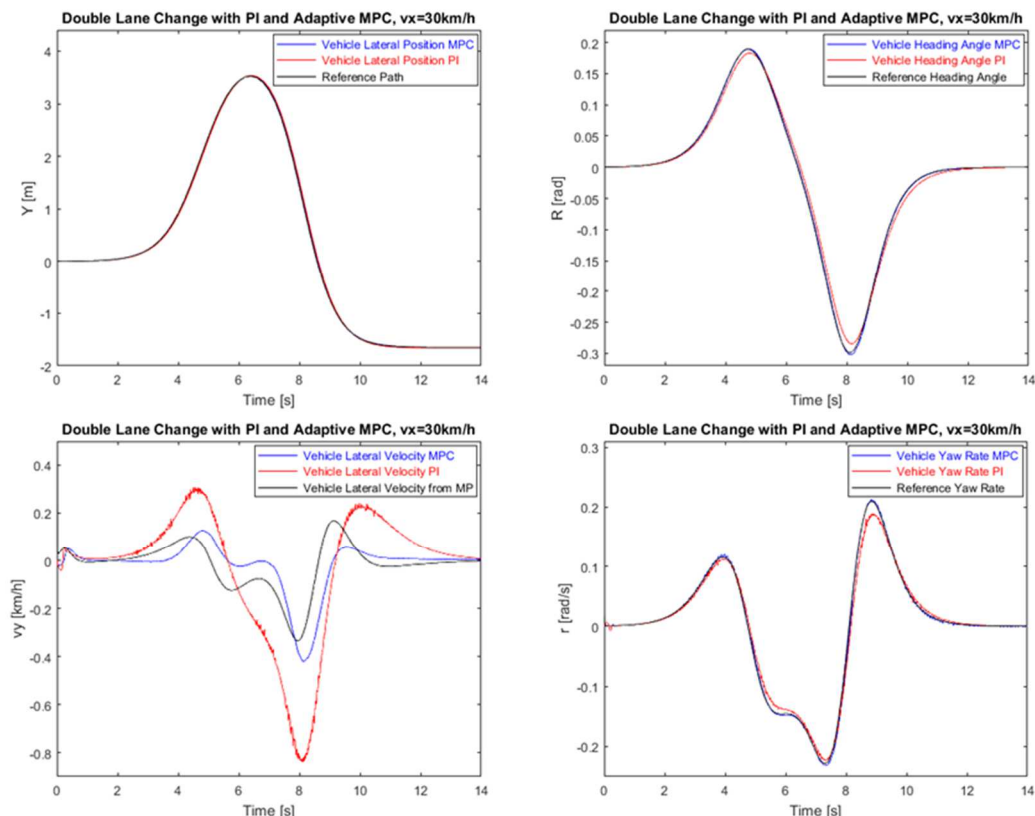


Figure 6-22 Comparison of PI and MPC Controller for double lane change at 30 km/h

Figure 6-22 shows the simulation results of the double lane change for 30 km/h. It can be seen, that the performance of the PI controller deteriorates, but can still maintain acceptable tracking of the reference signals, with the exception of  $v_y$ .

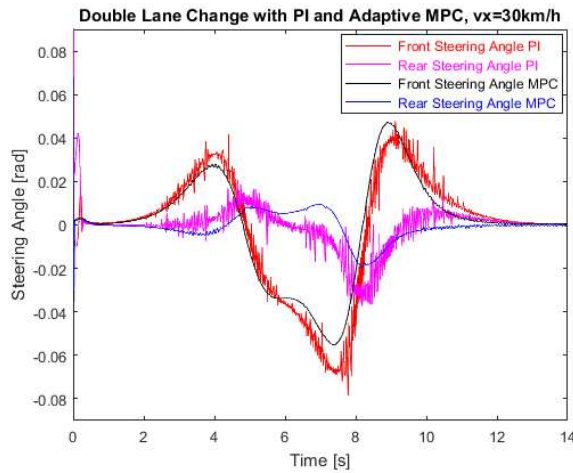


Figure 6-23 Comparison of computed steering angles at 30 km/h

In Figure 6-23 the steering angles of front and rear wheel computed by MPC and PI controller are compared. The signal quality of the PI steering angles worsened with the increasing speed, whereas the MPC controller still computes smooth and feasible steering angles.

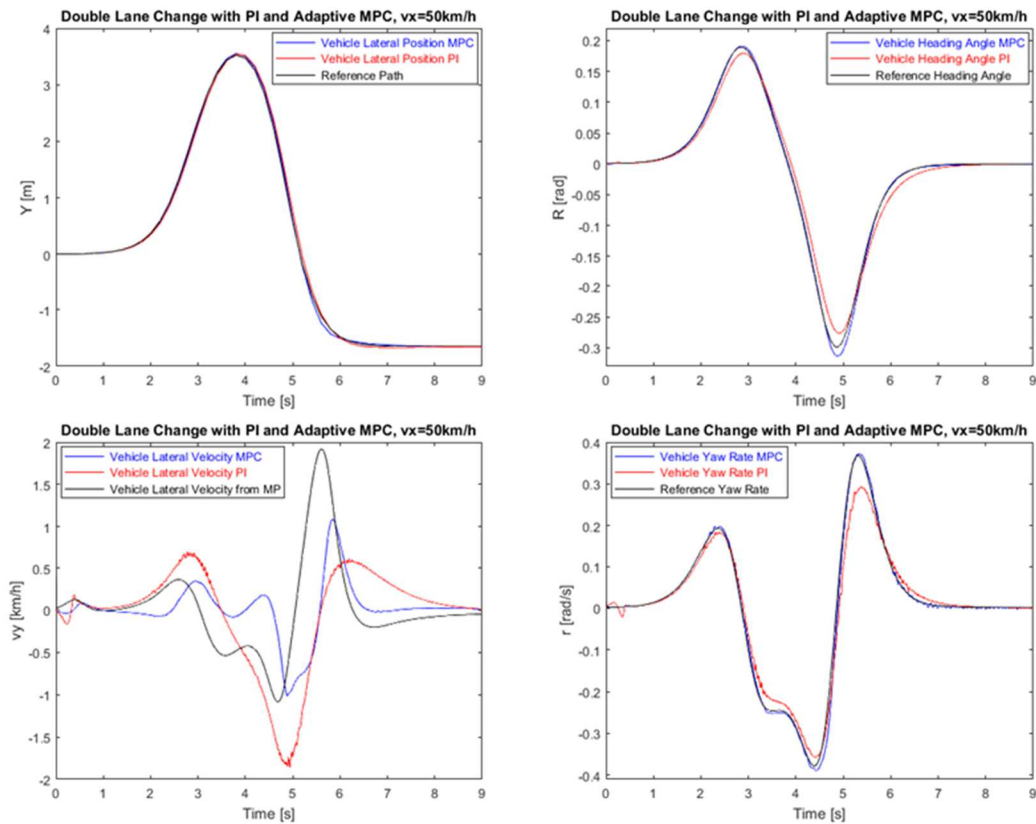
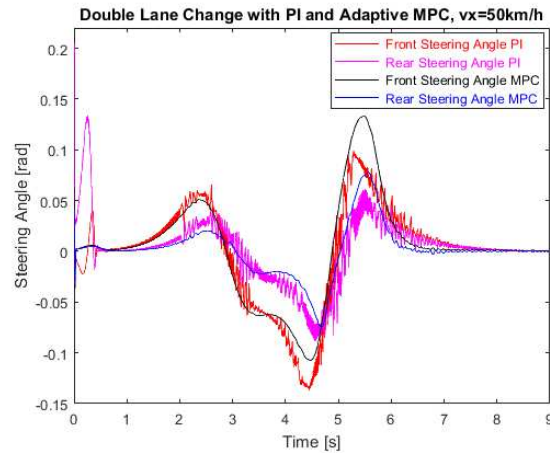


Figure 6-24 Comparison of PI and MPC Controller for double lane change at 50 km/h

The final simulation of the MPC Steering Controller and the PI Steering Controller was performed at 50 km/h and can be seen in Figure 6-24. At this point, the performance of the

MPC controller deteriorates a bit and small deviations from the reference signals can be seen. The PI, on the other hand, shows higher deviations, by basically still maintaining the trajectory following. The lateral vehicle speed  $v_y$  is not matched by both controllers, which might root from the drawbacks of using a kinematic vehicle model in the MPC Motion Planner, as discussed in Chapter 6.3.2.



**Figure 6-25** Comparison of computed steering angles at 50 km/h

The steering angles depicted in Figure 6-25 from the double lane change simulation at 50 km/h show a similar behavior as the ones from the 30 km/h simulation. The fluctuations are high and considering a real application in a vehicle these signals can't be used for steering since the actuators won't be able to perform the rapid changes. The MPC Steering Controller maintains feasible control signals.

Concluding from the conducted simulations it can be said that the PI Steering Controller is in contrast to the MPC Steering Controller not suitable for controlling a nonlinear vehicle dynamics model. The MPC controller is able to take constraints on both signal bounds and changing rates into accounts and maintain proper control signals over a range of different prediction model parameters, like the vehicle longitudinal speed  $v_x$  and the PI Controller used in this thesis failed in gaining the same result. Furthermore, only one MPC controller was used in this example, whereas two PI controllers were necessary. This lies in the MPC nature of being able to handle multiple inputs and multiple outputs at the same time. A further aspect is that the PI controller showed high sensitivity to changes in vehicle longitudinal speed  $v_x$ , which the MPC controller can handle thanks to its adaptive design.

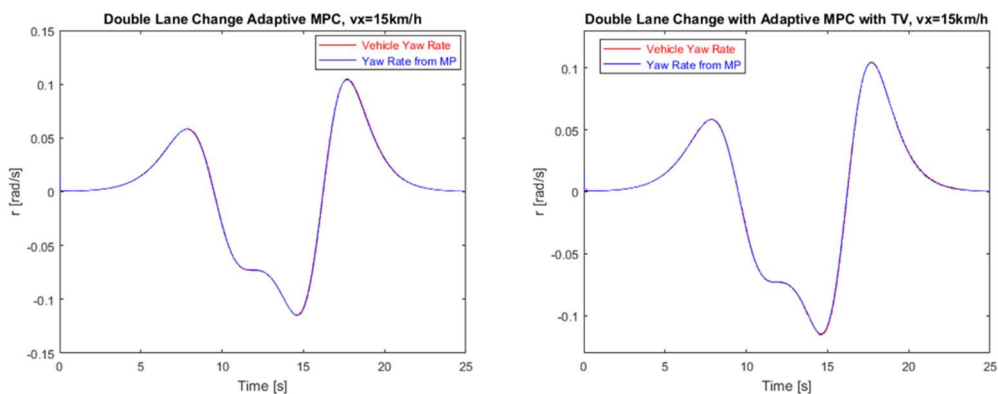
In addition to the simulation results using a PI controller, several simulations with PID and PD controllers were conducted using the same control structure as presented in Figure 6-19. All of these variants were tuned by using both the "T-Sum Rule" and the tuning provided by the Simulink® software. The simulations were again performed at 15 km/h, 30 km/h and 50 km/h and compared with the results gained from the MPC controllers. It turned out that all these configurations showed similar results as the previously described PI controller considering the tracking performance and the steering angle computation.

## 6.5 Simulation of MPC Torque Vectoring Controller

The MPC Torque Vectoring Controller introduced in Chapter 4.3 can be used in all driving modes, which means that it should provide additional support in trajectory following when four-wheel steering mode is applied and taking over the whole vehicle control in case of sideways driving and turning on the spot. In this chapter, all three driving modes are simulated with the Torque Vectoring Controller.

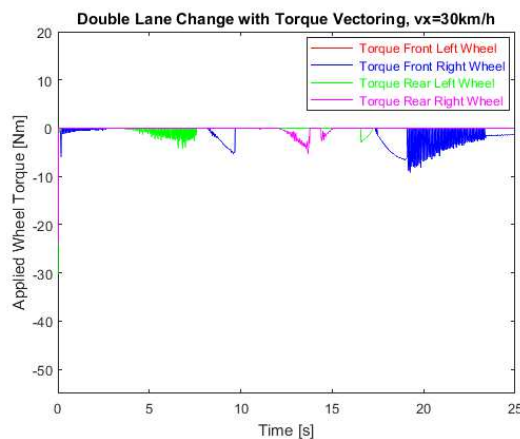
### 6.5.1 Double Lane Change with and without Torque Vectoring

In Chapter 6.3.2 the double lane change manoeuvre is simulated at various speeds by using Adaptive MPC controllers for Motion Planning and Steering Control. The results from these simulations are compared with the results when the MPC Torque Vectoring Controller is used additionally. Torque vectoring mainly influences vehicle yaw rate  $r$ , as discussed in Chapter 4.3. Therefore, yaw rate change during simulation is assessed.



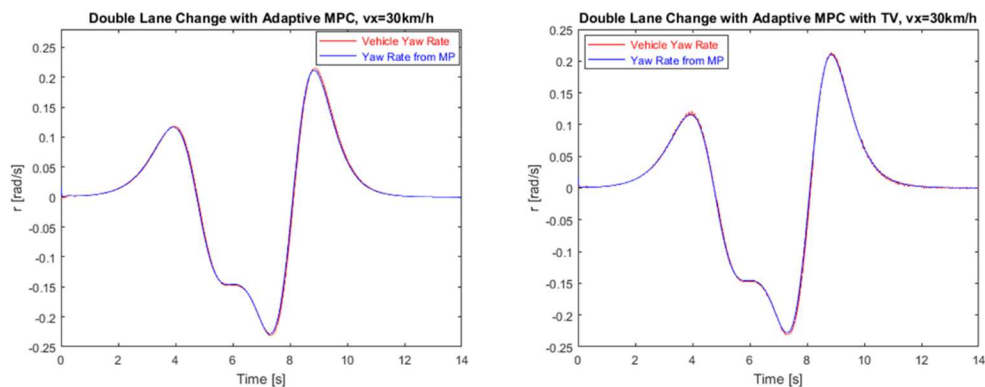
**Figure 6-26** Yaw rate during double lane change without and with Torque Vectoring at 15 km/h

Figure 6-26 shows the yaw rate of the vehicle during double lane change at  $v_x = 15\text{ km/h}$ . In the left graph, no torque vectoring was applied and it can be seen that the yaw rate of the vehicle matches well with the reference signal. Therefore, the right picture with torque vectoring shows no difference to the one without.



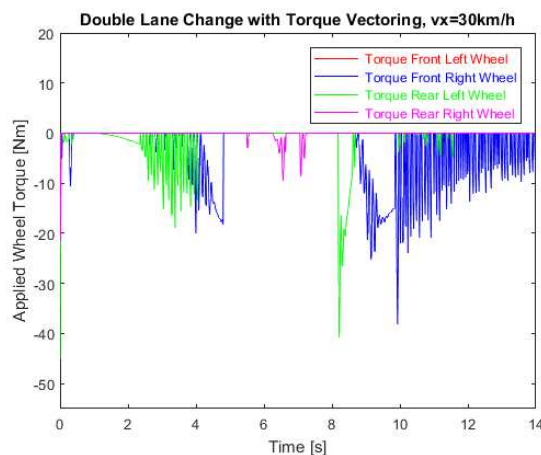
**Figure 6-27** Torque inputs at 15 km/h

The corrective braking inputs of the MPC Torque Vectoring Controller can be seen in Figure 6-27. Compared with Figure 6-26 the corrective inputs correlate with the dynamic regions of the manoeuvre.



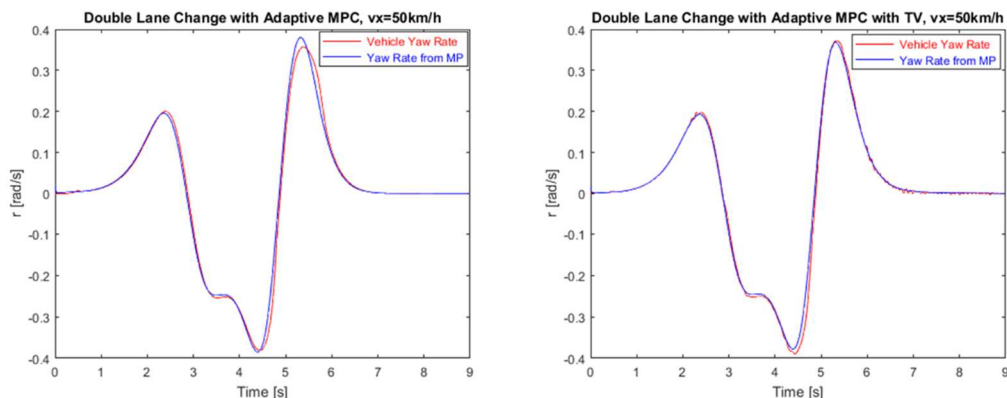
**Figure 6-28** Yaw rate during double lane change without and with Torque Vectoring at 30 km/h

The results in Figure 6-28 were obtained by simulating the double lane change at a speed of  $v_x = 30\text{km/h}$ . The right graph shows the resulting vehicle yaw rate when torque vectoring is applied. One can see, that the vehicle yaw rate follows the reference signal more closely than it is the case in the left picture without torque vectoring.



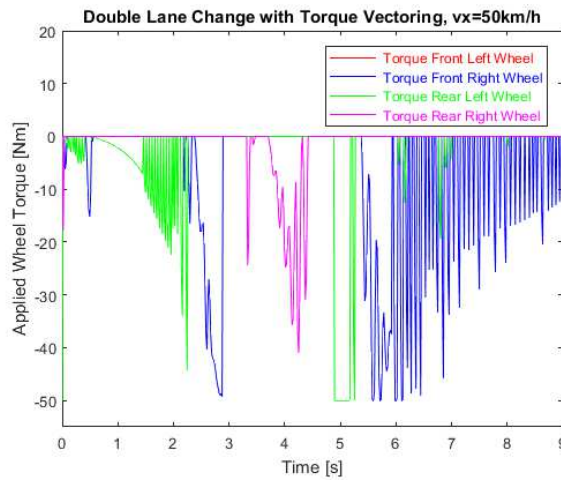
**Figure 6-29** Torque inputs at 30 km/h

Figure 6-29 shows the torque vectoring signals to the vehicle, which were applied during the double lane change at 30km/h. Compared to Figure 6-27 the torque values are higher, which comes in line with the higher vehicle dynamics.



**Figure 6-30** Yaw rate during double lane change without and with Torque Vectoring at 50 km/h

In Figure 6-30 the simulation results for a double lane change at  $v_x = 50\text{km/h}$  are displayed. The left graph shows the results without and the right graph with torque vectoring. Similar to the results from the simulation at 30 km/h, the vehicle yaw rate follows the reference signal more accurate when torque vectoring is used. Especially in the regions with high yaw rate and rapid change of direction, the MPC Torque Vectoring Controller can achieve an improvement in yaw rate tracking compared to the results without torque vectoring. This results in a better vehicle stabilization at higher speeds and therefore improves the usability of the whole control structure.



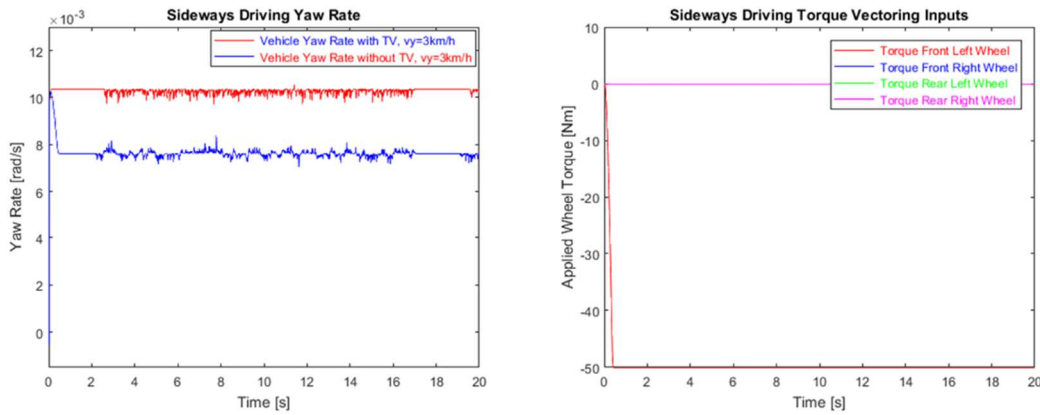
**Figure 6-31** Torque inputs at 50 km/h

The double lane change at 50km/h already shows significant vehicle dynamics. The braking torque applied to the vehicle model is, therefore, higher than in the two simulations before as it can be seen in Figure 6-31. Sometimes the braking torque even reaches the constrained limit but stays below that most of the time.

### 6.5.2 Sideways Driving

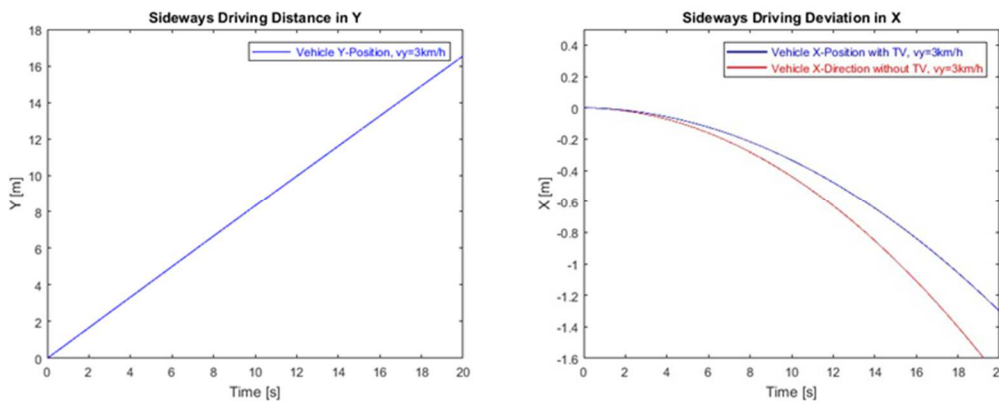
In case of sideways driving the vehicle travels in lateral vehicle direction with defined steering angles. As described in Chapter 4.3.1 the only corrective input to follow a defined lateral trajectory is to brake certain wheels with the MPC Torque Vectoring Controller. The suitability of this idea is evaluated in this chapter.

In order to simulate sideways driving where a corrective torque vectoring is needed, a scenario has been chosen, which can easily happen in a real application. This refers to a poor setting of the initial steering angles, which could also happen due to mechanical clearance in the steering system. Therefore, the steering angle of the front-right wheel is set as  $89^\circ$  and the one of the rear left wheel to  $91^\circ$ . Based on these boundary conditions, a simulation of sideways driving at vehicle lateral speed  $v_y = 3\text{km/h}$  is performed.



**Figure 6-32** Yaw rate and braking torque during sideways driving with steering angle mismatch

Figure 6-32 shows the vehicle yaw rate  $r$  and the torque inputs from the MPC Torque Vectoring Controller. It can be seen that the torque signal immediately rises up to the biggest braking torque, which was set as  $-50\text{Nm}$  in the constraints. Due to this the vehicle yaw rate in the left picture caused by the torque vectoring controller is held on a constant level in order to counteract the steering angle error.



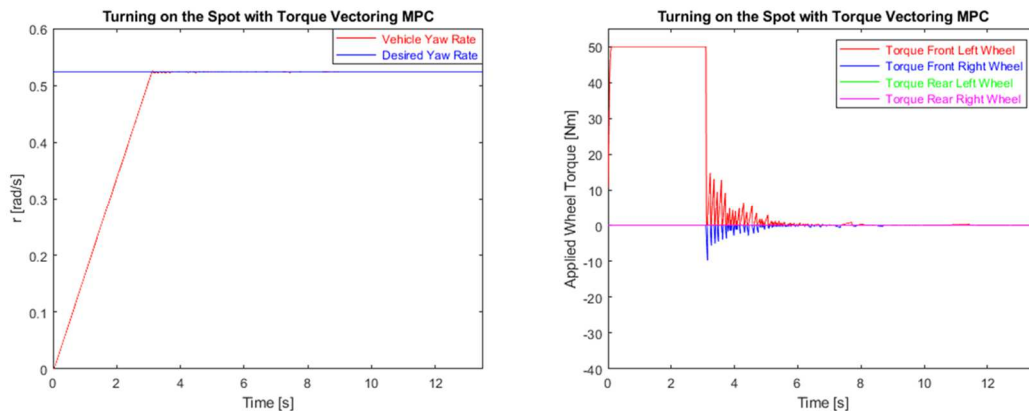
**Figure 6-33** Lateral and longitudinal position during sideways driving with steering angle mismatch

The results presented in Figure 6-33 show the path in global  $Y$  direction the vehicle traveled during the simulation and the path in the global  $X$  direction, which should be zero but is caused by the steering angle errors. It can be seen that the error in the global  $X$  direction is lowered when the MPC Torque Vectoring Controller is used.

From this simulation it can be seen that torque vectoring has an effect in sideways driving and can possibly reduce errors in the trajectory. But it also shows by the high applied braking torque that the limits of this strategy are reached in this case. Therefore, the conclusion is that the MPC Torque Vectoring Controller can help to reduce errors in sideways driving but not fully compensate them. The reason for that is that torque vectoring cannot replace a steering system when it comes to correcting trajectories. Another aspect is that the motion of the vehicle when drifting off in longitudinal direction during sideways driving occurs mainly in lateral tire direction. Therefore, the tire characteristics in lateral direction influence this motion strongly, e.g. the lateral slip and the cornering stiffness, as described in Chapter 4.3.1.

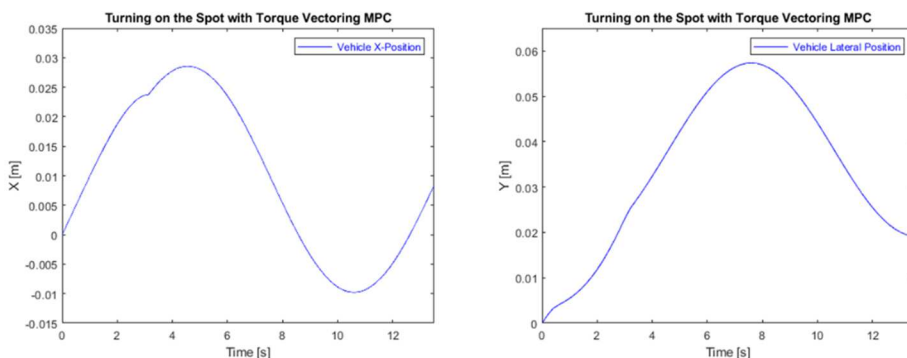
### 6.5.3 Turning on the Spot

The driving mode for turning on the spot was described in Chapter 3.2.2 and 4.3.1 and only the MPC Torque Vectoring Controller can be used since the steering angles are predefined and therefore, an obstacle avoidance manoeuvre, which would require the Motion Planner can't be performed. In order to simulate the performance of the torque vectoring controller, a constant reference yaw rate is defined, which the vehicle must follow. As mentioned in Chapter 4.3.2, it is necessary to set a small vehicle longitudinal velocity e.g.  $v_x = 0,01 \text{ m/s}$  (or  $0,036 \text{ km/h}$ ) in order to avoid dividing by zero in the state matrices, even though both longitudinal and lateral velocity should be zero during this manoeuvre. The influence of this intentionally caused error will be shown in this chapter.



**Figure 6-34** Yaw rate and braking torque during turning on the spot

Figure 6-34 shows the yaw rate and torque vectoring signals of a turning on the spot manoeuvre. It has to be mentioned, that the simulation is started with a vehicle yaw rate of  $r = 0$ . This causes the torque vectoring controller to apply the maximum driving torque on a wheel in order to reach the desired yaw rate. This part of the manoeuvre should normally be done by a slip controller, which is used to drive the wheels and the torque vectoring controller should only take actions when there are deviations from the reference signal. Furthermore, the torque vectoring controller is not suitable for driving the vehicle because it doesn't monitor the wheel slip and can therefore not prevent wheel spin when driving or lock when braking. The curves shown in Figure 6-34 are the only representative for showing the torque vectoring performance after roughly three seconds of simulation time when the vehicle yaw rate is already close to the reference value. After that, the torque vectoring principle applies braking and driving torques until the desired reference yaw rate can be kept almost constant.



**Figure 6-35** Longitudinal and lateral position during turning on the spot

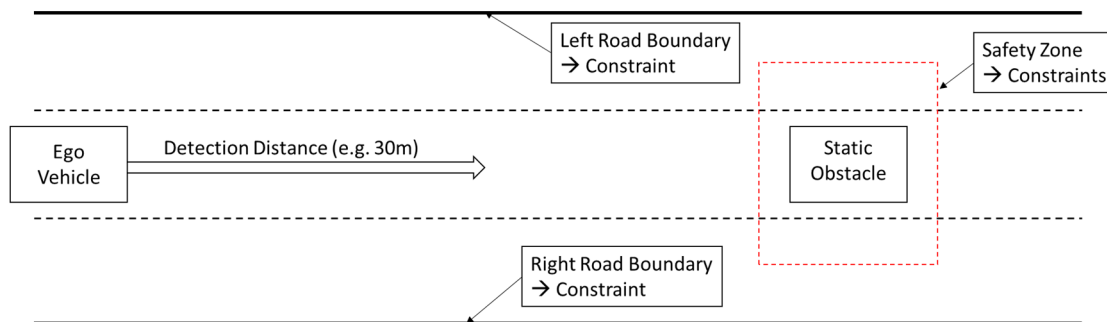


The curves shown in Figure 6-35 describe the displacement of the vehicle in global  $X$  and  $Y$  direction during the turning on the spot manoeuver. This displacement is caused by the vehicle longitudinal speed  $v_x = 0,01m/s$  and it can be seen that the effect on the vehicle position is marginal and could be compensated in the navigation level.

## 6.6 Simulation of Obstacle Avoidance

### 6.6.1 Obstacle Avoidance Scenario

In autonomous driving, it is inevitable to have a system on board, which allows adjusting the vehicle trajectory in order to avoid colliding with objects in the vehicle environment and therefore ensure safe driving. The field of obstacle avoidance is very broad since many different kinds of obstacles can be distinguished and with them come different avoidance actions. The focus of this thesis was to establish and simulate Model Predictive Controllers for controlling a nonlinear vehicle model. Therefore, only a simplified obstacle avoidance scenario is applied in order to assess the suitability of the Motion Planner in that field. The example used in this simulation is derived and adapted from [29].



**Figure 6-36** Obstacle Avoidance Scenario

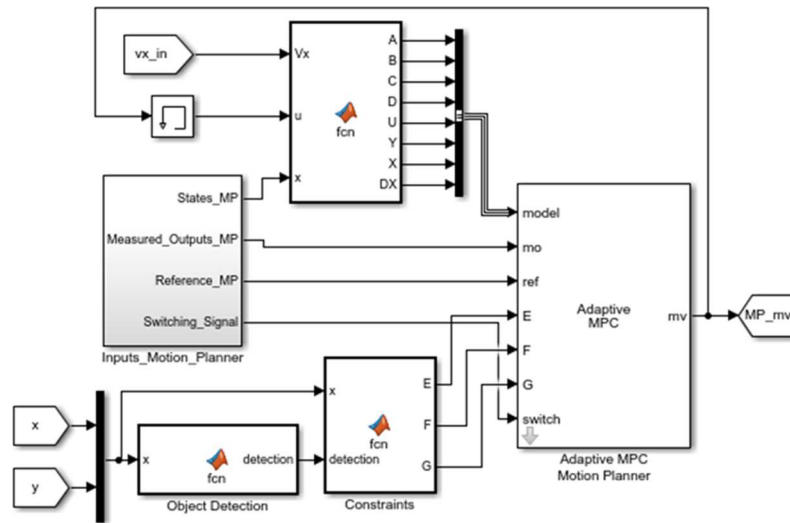
Figure 6-36 shows an obstacle avoiding scenario considering a static obstacle on a three-lane road, with the dashed safety zone around the obstacle. This safety zone is necessary because only the motion of the vehicle center of gravity is computed, not representing the full vehicle size. In order to ensure passing by the obstacle, the size of it has to be increased. If the center of gravity of the vehicle then travels along the constrained safety zone, a collision is avoided. Besides the obstacle shape, the left and right road boundaries are defined as constraints for lateral movement. These constraints are updated at each control interval and represent the area in which the vehicle can drive to avoid a collision with the obstacle.

The constraints that are used in this case apply on both input and output variables and are therefore called mixed input/output constraints.

$$Eu + Fy \leq G \quad (6.3)$$

Equation (6.3) shows the mixed input/output constraints, with the constraint matrices  $E$ ,  $F$  and  $G$ , the input vector  $u$  and the output vector  $y$ . The constraint matrix  $G$  can, for example, hold the upper and lower road boundaries as limitations for sideways movement during normal driving [59].

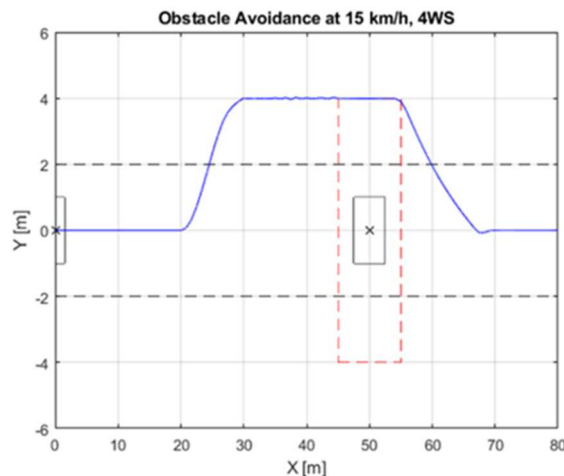
### 6.6.2 Implementation in Simulink®



**Figure 6-37** Adaptive MPC Motion Planner for Obstacle Avoidance

Figure 6-37 shows the Motion Planner with obstacle avoidance in Simulink, as introduced in Chapter 5.4. An Object Detection function receives the current vehicle position in global  $X$  and  $Y$  position as feedback from the vehicle model and the sensor information, which gives information about the obstacle position as soon as it is within the sensor detection range. If no obstacle is detected the Constraints function sends the standard constraints to the Adaptive MPC Controller, for example the road boundaries. In case an obstacle is detected, special constraints are computed, which include the distance to the obstacle and the safety zone around it. The vehicle can then move around the obstacle considering the constraints.

### 6.6.3 Results and Discussion



**Figure 6-38** Resulting vehicle path for obstacle avoidance

Figure 6-38 shows the path the vehicle takes around the static obstacle during the obstacle avoidance manoeuvre at a constant vehicle longitudinal speed of  $v_x = 15\text{km/h}$ . It can be seen that the vehicle clearly passes by the obstacle without violating the safety zone and after passing, it goes back to the center of the middle lane. This example shows, that the designed MPC Motion Planner is basically able to perform an obstacle avoidance manoeuvre.

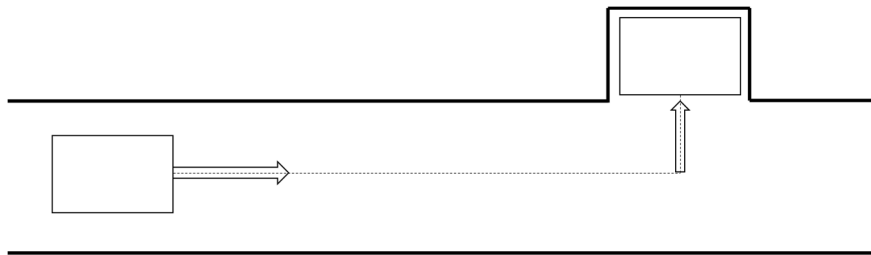
An obstacle avoidance manoeuvre can only be done in the four-wheel steering mode. In case of sideways driving or turning on the spot the steering angles are fixed and therefore

steering around an obstacle is not possible. In these modes, only a collision avoidance can be achieved by stopping the vehicle driving or turning motion before a collision occurs.

## 6.7 Simulation of Combined Modes

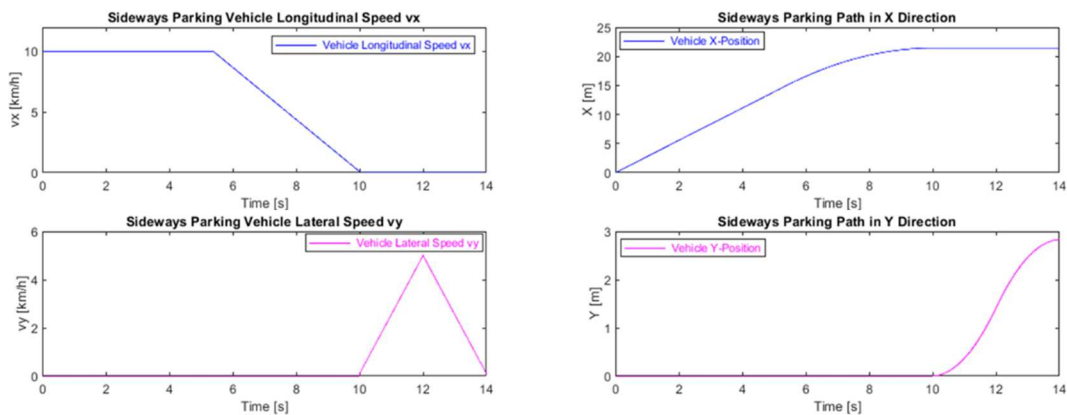
### 6.7.1 Parking Scenario

This simulation scenario combines the mode of four-wheel steering and sideways driving, which should represent a parking scenario.



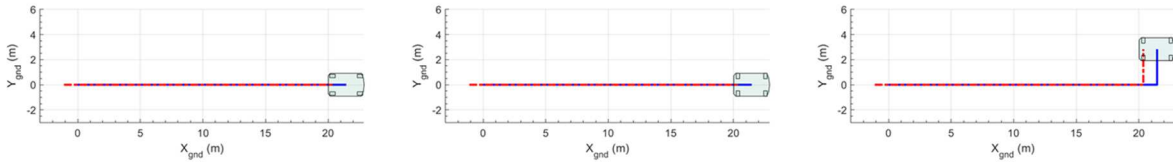
**Figure 6-39** Sideways parking scenario

Figure 6-39 shows the sideways parking scenario, which combines straight-ahead driving with sideways driving. The parking spot is just a bit larger than the vehicle itself and therefore there is no other way of parking than to use sideways driving.



**Figure 6-40** Input signals and results of sideways parking manoeuver

The control inputs for the combined sideways parking manoeuver are shown in the left half of Figure 6-40, where the vehicle longitudinal speed  $v_x$  decreases from a constant initial value when the parking spot is reached and settles to zero when forward driving is switched off. After that, the speed ramp of lateral velocity  $v_y$  drives the vehicle in the lateral direction. The graphs on the right side of Figure 6-40 show the resulting trajectories of the vehicle in global  $X$  and  $Y$  direction.

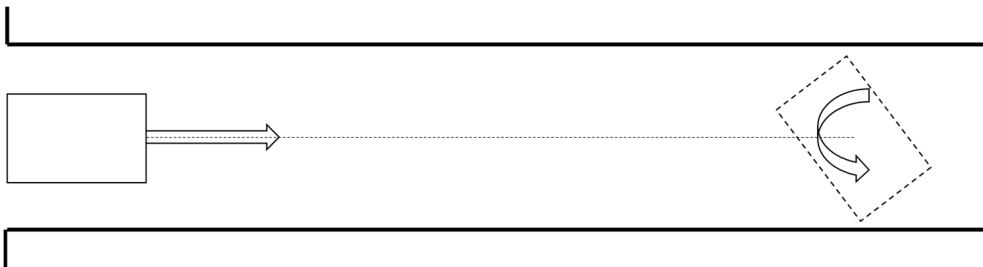


**Figure 6-41** Visualization of the sideways parking scenario

Figure 6-41 shows the visualization of the sideways parking manoeuvre. The first picture on the left shows the vehicle as it has reached the parking spot. At this point a mode signal switches off the MPC Motion Planner and the MPC Steering Controller because they are not needed for sideways driving. This mode signal also initializes the steering angles for sideways driving, which can be seen in the middle picture. The picture on the right shows the vehicle in its final parking position.

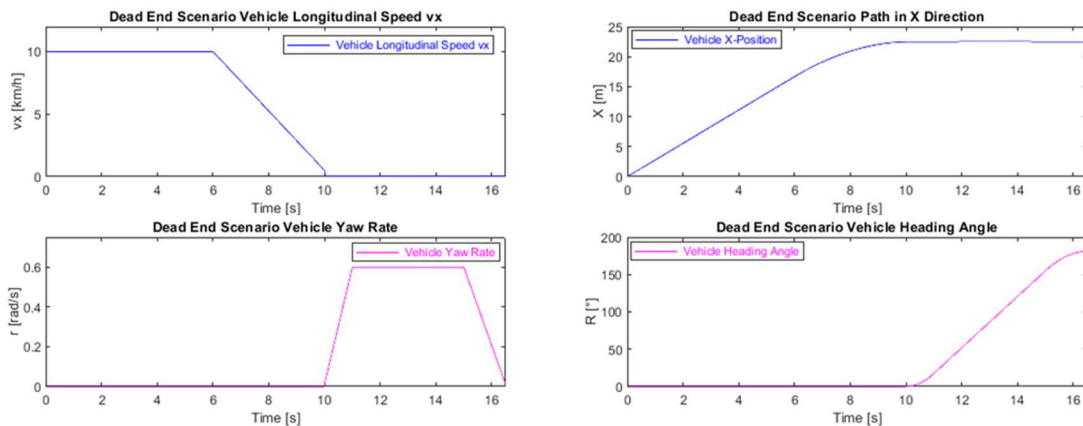
### 6.7.2 Dead-End Scenario

The dead end scenario represents a combination of the four-wheel steering mode and turning on the spot.



**Figure 6-42** Dead end scenario

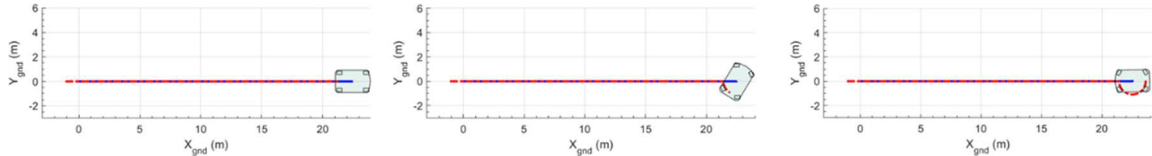
Figure 6-42 describes the dead-end scenario. The task is to turn the vehicle around at the end of the street. The reason for that could be that the vehicle has to drive out of the dead



**Figure 6-43** Input signals and results of the dead-end scenario

end with the front end first or it could be that the dead end is a loading bay and the goods will be placed on the rear side of the vehicle.

The graphs in Figure 6-43 show the input signals of the dead-end scenario and the resulting path. As can be seen in the top left picture, the vehicle slows down by the end of the street. After the mode signal triggered the switching off of MPC Motion Planner and MPC Steering Controller, the steering wheels are set according to the turning on the spot mode and the yaw rate signal shown in the bottom left part of Figure 6-43 is applied to the vehicle. It goes back to zero in order to gain the wanted turning angle of 180°. The results of the global X position and vehicle heading angle  $\theta$  can be seen in the right part of Figure 6-43. The heading angle  $\theta$  rises until 180° and therefore puts the vehicle in the right orientation to be loaded or to leave the dead-end.

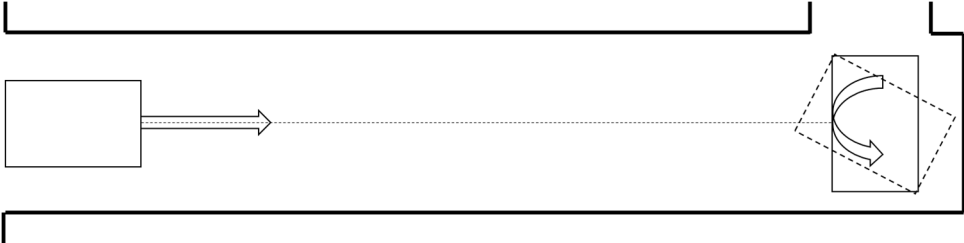


**Figure 6-44** Visualization of the dead end manoeuver

The visualization of the dead end scenario in Figure 6-44 shows some of the steps during the manoeuver. The first picture shows the vehicle when it has reached its final point in the dead-end. Then the mode is switched to turning on the spot and the four steering angles are set according to the turning on the spot mode. The second picture shows the vehicle during turning with the wheels arranged for turning on the spot. In the final picture, the vehicle completed a 180° turn.

6.7.3 Right Angle Turn

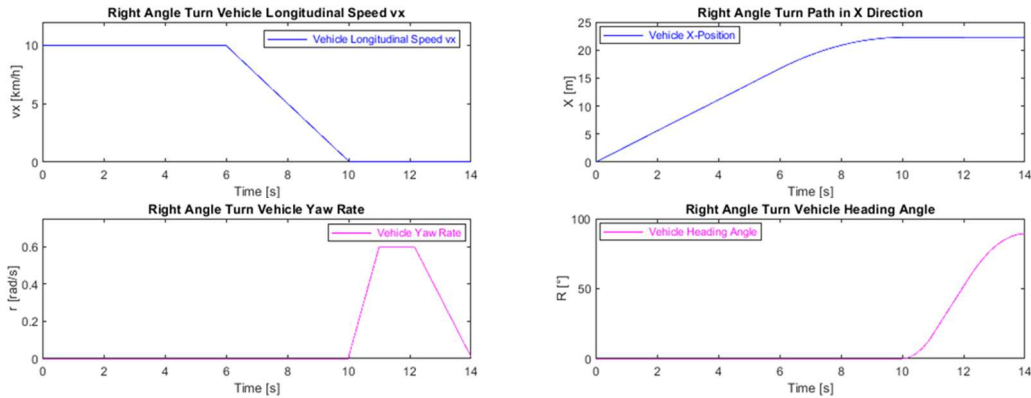
The right angle turn is another example of the combination of four-wheel driving mode and turning on the spot.



**Figure 6-45** Right angle turn scenario

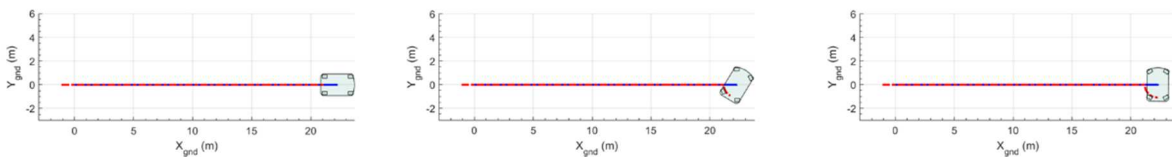
The scenario in Figure 6-45 can be seen as a case where the high maneuverability of the 4WID-4WIS vehicle is utilized. At the end of the road, the vehicle should take a corner, which is too narrow for normal turning and the gap is too small for sideways driving. Performing

a 90° turn can orientate the vehicle in a way that it can drive on straight ahead into the narrow road and possibly fulfill a delivery task.



**Figure 6-46** Input signals and results for right angle turn manoeuver

The input signals and resulting trajectories depicted in Figure 6-46 are quite similar to the ones of the dead-end scenario. The difference lies in the duration the yaw rate is applied because the vehicle should only do a 90° turn. The achievement of the desired heading angle can be seen in the bottom right graph in Figure 6-46.



**Figure 6-47** Visualization of the right angle turn manoeuver

In Figure 6-47 some simulation steps of the right angle turn manoeuver are shown. After the vehicle has reached the point at which it should turn by using the four-wheel steering mode, a switching signal is sent to the controllers in order to turn off the optimization in both Motion Planner and Steering Controller and to adjust the wheels with the required steering angles. These steering angles can be seen in the middle picture as the vehicle is turning on the spot. In the right picture, a 90° heading angle of the vehicle is reached and the manoeuver is completed.

## **7 Conclusion and Outlook**

### **7.1 Conclusion**

Considering the main research content described in Chapter 1.4, several tasks had to be accomplished. In terms of controller design, the basic vehicle models were described mathematically and the checking for their suitability was successful. Two different controller types were tested during which it turned out that the Adaptive MPC is a good choice for the considered vehicle control task. This is because it shows a certain robustness against changes in vehicle longitudinal speed and can control the nonlinear vehicle model even at higher speeds. The task of trajectory tracking was accomplished by the introduced MPC Steering Controller and refined by the use of the MPC Torque Vectoring Controller. Motion planning for obstacle avoidance was shown to be possible with the designed MPC Motion Planner and therefore accomplished one of the predefined tasks. For realizing different driving manoeuvres, the possible approaches were discussed and simulated, showing promising and satisfactory results even for the combination of modes.

The conclusion is that the designed control strategy is suitable for the use with a nonlinear vehicle model for the considered field of application, fulfilling some basic requirements of autonomous driving. This basis can be used in a real vehicle prototype after further development.

### **7.2 The Direction of Further Work**

During this thesis, some simplifications were made, which turned out to be acceptable during the conducted simulations, but a possible ongoing research in that field could include to test the controllers outside their designed field in order to ensure practicability for real use.

Since the focus of this thesis was to enable trajectory following, obstacle avoidance and special driving modes, the propulsion of the vehicle was not taken into account. This topic represents a future field of research by implementing Model Predictive Control for driving the wheels and enabling safe and stable driving on various road conditions.

Due to related projects in the department, the Motion Planner was designed to compute velocity commands from a predefined path. This path is provided by the Navigation Level and an interesting future work is to compute a collision-free trajectory within the Motion Planning level using the MPC control strategy.

## References

- [1] B. Paden, M. Cap, S. Zheng Yong, D. Yershov and E. Frazzoli, "A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles," *IEEE TRANSACTIONS ON INTELLIGENT VEHICLES*, VOL. 1, NO. 1,, MARCH 2016.
- [2] SAE International, "SAE International," 20 09 2016. [Online]. Available: <http://articles.sae.org/15021/>. [Accessed 08 12 2018].
- [3] Wikipedia, „J3016, SAE Standard,“ 15 11 2018. [Online]. Available: [https://de.wikipedia.org/wiki/SAE\\_J3016](https://de.wikipedia.org/wiki/SAE_J3016). [Zugriff am 07 12 2018].
- [4] D. Christofides, "Control of Nonlinear Distributed Process Systems: Recent Developments and Challenges," *AIChE Journal*, vol. 47, no. 3, p. 514, 2001.
- [5] N. H. Amer, H. Zamzuri and K. Hudha, "Modelling and Control Strategies in Path Tracking Control for Autonomous Ground Vehicles: A Review of State of the Art and Challenges," *J Intell Robot Syst (2017)* 86, pp. 225-254.
- [6] F. Borelli, P. Falcone, T. Keviczky, J. Asgari and D. Hrovat, "MPC-Based Approach to Active Steering for Autonomous Vehicle Systems," *International Journal of Vehicle Autonomous Systems* 3(2), pp. 265-291, 2005.
- [7] H. B. Pacejka, *Tyre and Vehicle Dynamics*, Second Edition, Elsevier, 2006: 172-184.
- [8] J. Adamy, *Nichtlineare Systeme und Regelungen*, 3. Auflage, Springer Vieweg, 2018: 519-520.
- [9] Y. D. Setiawan, T. H. Nguyen and P. S. Pratama, "Path Tracking Controller Design of Four Wheel Independent Steering Automatic Guided Vehicle," *International Journal of Control, Automation and Systems* 14(6), pp. 1550-1560, 2016.
- [10] W. Zhao, Y. Li and C. Wang, "H<sup>∞</sup> control of novel active steering integrated with electric power steering function," *Journal of Central South University*, no. 20, p. 2151–2157, 2013.
- [11] W. Zhao and X. Qin, "Study on mixed H<sub>2</sub>/H<sup>∞</sup> robust control strategy of four wheel steering system," *Science China Technological Sciences*, no. 60, p. 1831–1840, 2017.
- [12] A. Eskandarian, *Handbook of Intelligent Vehicles*, Volume 2: 85-87, Springer, 2012.
- [13] J. K. Fredlund and K. S. Sulejmanovic, "Autonomous driving using Model Predictive Control methods, MSc Thesis," 2017. [Online]. Available: <http://lup.lub.lu.se/luur/download?func=downloadFile&recordId=8906188&fileId=8906189>. [Accessed 07 12 2018].
- [14] B. Kouvaritakis and M. Cannon, *Model Predictive Control - Classical, Robust and Stochastic: 1-5*, Springer , 2016.



- C. E. Beal, "Applications of model predictive control to vehicle dynamics for active safety and stability," 2011. [Online]. Available: <https://purl.stanford.edu/rv794bn8004>. [Accessed 07 12 2018].
- [15]
- P. Hang, X. Chen and F. Luo, "Path-Tracking Controller Design for a 4WIS and 4WID Electric Vehicle with Steer-by-Wire System," *SAE Technical Paper 2017-01-1954*, 2017.
- [16]
- MathWorks®, "MPC Modeling," [Online]. Available: [https://www.mathworks.com/help/releases/R2018a/mpc/gs/mpc-modeling.html?searchHighlight=MPC%20Modeling&s\\_tid=doc\\_srchtile](https://www.mathworks.com/help/releases/R2018a/mpc/gs/mpc-modeling.html?searchHighlight=MPC%20Modeling&s_tid=doc_srchtile). [Accessed 07 12 2018].
- [17]
- D. Q. Mayne, J. B. Rawlings, C. V. Rao and P. O. M. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica, Volume 36, Issue 6*, pp. 789-814, 06 2000.
- [18]
- MathWorks®, "QP Solver," [Online]. Available: [https://www.mathworks.com/help/releases/R2018a/mpc/ug/qp-solver.html?searchHighlight=mpc%20solver&s\\_tid=doc\\_srchtile](https://www.mathworks.com/help/releases/R2018a/mpc/ug/qp-solver.html?searchHighlight=mpc%20solver&s_tid=doc_srchtile). [Accessed 07 12 2018].
- [19]
- X. Qian, "Model predictive control for autonomous and cooperative driving, PhD Thesis," 14 11 2017. [Online]. Available: <https://pastel.archives-ouvertes.fr/tel-01635261>. [Accessed 07 12 2018].
- [20]
- Y. Gao, „Model Predictive Control for Autonomous and Semiautonomous Vehicles, PhD Thesis,“ 2014. [Online]. Available: <https://escholarship.org/content/qt8xd0b56h/qt8xd0b56h.pdf>. [Zugriff am 07 12 2018].
- [21]
- M. Nolte, M. Rose, T. Stolte and M. Maurer, "Model Predictive Control Based Trajectory Generation for Autonomous Vehicles – An Architectural Approach," *IEEE Intelligent Vehicles Symposium (IV)*, 2017.
- [22]
- R. Matthaei and M. Maurer, "Autonomous driving – a top-down-approach," *Automatisierungstechnik 2015; 63(3)*, p. 155–167.
- [23]
- R. N. Jazar, *Vehicle Dynamics - Theory and Application*, Third Edition: 409-421;565-603, Springer, 2008.
- [24]
- K. N. Spentzas, I. Alkhazali and M. Demic, "Kinematics of four-wheel-steering vehicles," *Forschung auf dem Gebiete des Ingenieurwesens 66*, pp. 211-216, 05 2001.
- [25]
- C. Liu, S. Lee , S. Varnhagen and E. . H. Tseng , "Path Planning for Autonomous Vehicles using Model Predictive Control," *IEEE Intelligent Vehicles Symposium (IV)*, 06 2017.
- [26]
- A. De Luca, G. Oriolo and C. Samson, "Feedback Control of a Nonholonomic Car-like Robot," in *Robot Motion Planning and Control*, Springer, p. Chapter 4.
- [27]

- 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, 5 2017.
- MathWorks, "Obstacle Avoidance Using Adaptive Model Predictive Control," [Online]. Available: <https://www.mathworks.com/examples/mpc/mw/mpc-ex91083107-obstacle-avoidance-using-adaptive-model-predictive-control>. [Accessed 07 12 2018].
- M. Babu, . Y. Oza, A. K. Singh, K. Madhava Krishna and S. Medasani, "Model Predictive Control for Autonomous Driving Based on Time Scaled Collision Cone," 12 2017. [Online]. Available: <https://arxiv.org/abs/1712.04965>. [Accessed 07 12 2018].
- A. Chakravarthy and D. Ghose, "Obstacle Avoidance in a Dynamic Environment: A Collision Cone Approach," *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS*, VOL. 28, NO. 5, pp. 562-574, 09 1998.
- Kiencke and L. Nielsen, *Automotive Control Systems - For Engine, Driveline, and Vehicle*: 303-304, Springer, 2005.
- R. Rajamani, *Vehicle Dynamics and Control*, Second Edition: 20-25, Springer, 2012.
- D. Schramm, M. Hiller and R. Bardini, *Vehicle Dynamics Modeling and Simulation*, Second Edition:225-231, Springer, 2018.
- W. Schwarting, J. Alonso-Mora, L. Paull, S. Karaman and D. Rus, "Parallel Autonomy in Automated Vehicles: Safe Motion Generation with Minimal Intervention," *IEEE International Conference on Robotics and Automation (ICRA)*, 05 2017.
- K. Zhang, J. Sprinkle and R. G. Sanfelice, "A Hybrid Model Predictive Controller for Path Planning and Path Following," *ICCPS '15 Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems*, pp. 139-148, 8 2015.
- P. Hang, F. Luo, S. Fang and X. Chen, "Path Tracking Control of a Four-Wheel-Independent-Steering Electric Vehicle based on Model Predictive Control," *Proceedings of the 36th Chinese Control Conference*, 7 2017.
- H. Zhou and Z. Liu, "Design of vehicle yaw stability controller based on model predictive control," *IEEE Intelligent Vehicles Symposium*, 2009.
- M. Choi and S. B. Choi, "MPC for vehicle lateral stability via differential braking and active front steering considering practical aspects," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 230(4), p. 459–469, 2016.
- E. Siampis, E. Velenis and S. Longo, "Model Predictive Torque Vectoring Control for Electric Vehicles Near the Limits of Handling," *European Control Conference (ECC), July 15-17, 2015. Linz, Austria, 2015*.

- G. Vasiljevic and S. Bogdan, "Model Predictive Control based Torque Vectoring Algorithm for Electric Car with Independent Drives," *24th Mediterranean Conference on Control and Automation (MED)*, 21-24 06 2016.
- [41]
- S. Breuer and A. Rohrbach-Kerl, *Fahrzeugdynamik - Mechanik des bewegten Fahrzeugs*: 57, Springer Vieweg, 2015.
- [42]
- G. Palmieri, O. Barbarisi, S. Scala and L. Glielmo, "An Integrated LTV-MPC Lateral Vehicle Dynamics Control: Simulation Results," in *Automotive Model Predictive Control*, Springer, 2010, pp. 231-255.
- [43]
- MathWorks, "State-Space Models, Part 1: Creation and Analysis," [Online]. Available: <https://www.mathworks.com/videos/state-space-models-part-1-creation-and-analysis-100815.html>. [Accessed 07 12 2018].
- [44]
- MathWorks, "State-Space Models, Part 2: Control Design," [Online]. Available: <https://www.mathworks.com/videos/state-space-models-part-2-control-design-100816.html>. [Accessed 07 12 2018].
- [45]
- MathWorks, "Design MPC Controller at the Command Line," [Online]. Available: <https://www.mathworks.com/help/mpc/gs/design-controller-using-the-command-line.html>. [Accessed 07 12 2018].
- [46]
- MathWorks, "Specify Scale Factors," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/scale-factors.html>. [Accessed 07 12 2018].
- [47]
- MathWorks, "Choose Sample Time and Horizons," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/choosing-sample-time-and-horizons.html>. [Accessed 07 12 2018].
- [48]
- MathWorks, „Understanding Model Predictive Control, Part 3: MPC Design Parameters,“ [Online]. Available: <https://www.mathworks.com/videos/understanding-model-predictive-control-part-3-mpc-design-parameters-1530607670393.html>. [Zugriff am 07 12 2018].
- [49]
- MathWorks, "How to Design Model Predictive Controllers," [Online]. Available: <https://www.mathworks.com/videos/how-to-design-model-predictive-controllers-1494253496907.html>. [Accessed 07 12 2018].
- [50]
- MathWorks, "Specify Constraints," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/specifying-constraints.html>. [Accessed 07 12 2018].
- [51]
- MathWorks, "review," [Online]. Available: <https://www.mathworks.com/help/mpc/ref/review.html>. [Accessed 07 12 2018].
- [52]
- MathWorks, "Adaptive MPC," [Online]. Available: <https://www.mathworks.com/help/mpc/ug/adaptive-mpc.html>. [Accessed 07 12 2018].
- [53]

- MathWorks, "Adaptive MPC Controller," [Online]. Available:
- [54] [https://www.mathworks.com/help/mpc/ref/adaptivempccontroller.html?searchHighlight=adaptive%20mpc%20simulink&s\\_tid=doc\\_srchtile](https://www.mathworks.com/help/mpc/ref/adaptivempccontroller.html?searchHighlight=adaptive%20mpc%20simulink&s_tid=doc_srchtile). [Accessed 07 12 2018].
- MathWorks, "Understanding Model Predictive Control, Part 7: Adaptive MPC Design with Simulink and Model Predictive Control Toolbox," [Online]. Available:
- [55] <https://www.mathworks.com/videos/understanding-model-predictive-control-part-7-adaptive-mpc-design-with-simulink-and-model-predictive-control-toolbox--1539762605717.html>. [Accessed 07 12 2018].
- MathWorks, "MPC Controller," [Online]. Available:
- [56] <https://www.mathworks.com/help/mpc/ref/mpccontroller.html>. [Accessed 07 12 2018].
- [57] R. Marino, S. Scalzi, G. Orlando and M. Netto, "A Nested PID Steering Control for Lane Keeping in Vision Based Autonomous Vehicles," *American Control Conference*, 2009.
- TU Graz, Institut für Regelungs- und Automatisierungstechnik, "Regelungstechnik I," [Online]. Available: <https://www.tugraz.at/institute/irt/lehre/ergaenzende-informationen/regelungstechnik-i/>. [Accessed 07 12 2018].
- MathWorks, "setconstraint," [Online]. Available:
- [59] <https://www.mathworks.com/help/releases/R2018a/mpc/ref/setconstraint.html>. [Accessed 07 12 2018].
- [60] K. Reif, *Brakes, Brake Control and Driver Assistance Systems: 19-21*, Wiesbaden: Springer Vieweg, 2014.
- G. Rill, "Vehicle Dynamics - Lecture Notes," October 2006. [Online]. Available:
- [61] <https://docplayer.net/61504505-Vehicle-dynamics-lecture-notes-prof-dr-georg-rill-october-2006.html>. [Accessed 03 01 2019].
- Z. Xiao, *Master Thesis: Four-wheel independent drive, independent steering Electric Vehicle: Vehicle Dynamics Modeling and Analysis for path control*, School of Automotive Studies, Tongji University Shanghai, 2017.

## 8 Appendix

Code of the MPC Controllers in MATLAB® Command Line:

```
% MPC for Motion Planner

vx=2.5;

Aa=[0 vx; 0 0];
Bb=[1 0; 0 1];
Cc=[1 0; 0 1];
Dd=zeros(2,2);
PlantMP=ss(Aa,Bb,Cc,Dd);

PlantMP.InputName={'vy','r'};
PlantMP.StateName={'Y','Theta'};
PlantMP.OutputName=PlantMP.StateName;
PlantMP.InputGroup.MV=2;
PlantMP.OutputGroup.MO=2;

isstable(PlantMP)
ctrb_matrix=ctrb(PlantMP)
rank(ctrb_matrix)

Ts1=0.017;

MPCobjMP=mpc(PlantMP,Ts1);

%Scale Factors
MPCobjMP.MV(1).ScaleFactor=100;           %for vy in m/s
MPCobjMP.MV(2).ScaleFactor=10;           %for r in deg/s
MPCobjMP.OV(1).ScaleFactor=30;           %for Y in m
MPCobjMP.OV(2).ScaleFactor=2*pi;         %for Theta in rad

%Prediction and Control Horizon
MPCobjMP.PredictionHorizon=20;
p1=MPCobjMP.PredictionHorizon;
MPCobjMP.ControlHorizon=4;

%Constraints
MPCobjMP.MV(1).Min=-5;                   %lower bound vy
MPCobjMP.MV(1).Max=5;                   %upper bound vy
MPCobjMP.MV(2).Min=-0.45;               %lower bound r
MPCobjMP.MV(2).Max=0.45;               %upper bound r
% MPCobjMP.MV(2).Min=0;                 %r for parallel steering
% MPCobjMP.MV(2).Max=0;                 %r for parallel steering

%Weights
MPCobjMP.Weights.OV(1)=20;              % 20: High Priority
MPCobjMP.Weights.OV(2)=10;             % 1: Average priority

ct_plant1=MPCobjMP.Model.Plant;
dt_plant1=c2d(ct_plant1,Ts1);
MPCobjMP.Model.Plant=dt_plant1;

review(MPCobjMP);

%% MPC for Trajectory Follower Steering Controller SC

m=500;
```

```

Iz=488;
Lf=1.05;
Lr=1.15;
g=9.81;

%values of tire cornering stiffness derived from simulation model
Caf=3.3469e+04;           %tire cornering stiffness in N/rad
Car=3.6656e+04;           %tire cornering stiffness in N/rad

%Plant Model Steering Controller
E1=(-Caf-Car)/(m*vx);
E2=((Lr*Car-Lf*Caf)/(m*vx))-vx;
E3=(Lr*Car-Lf*Caf)/(Iz*vx);
E4=(-Lf*Lf*Caf-Lr*Lr*Car)/(Iz*vx);
F1=(Caf/m);
F2=(Car/m);
F3=((Lf*Caf)/Iz);
F4=(-(Lr*Car)/Iz);

E=[E1 E2; E3 E4];
F=[F1 F2; F3 F4];
G=[1 0; 0 1];
H=[0 0; 0 0];
PlantSC=ss(E,F,G,H);

PlantSC.InputName={'Delta_f','Delta_r'};
PlantSC.StateName={'vy','r'};
PlantSC.OutputName=PlantSC.StateName;
PlantSC.InputGroup.MV=2;
PlantSC.OutputGroup.MO=2;

Ts2=0.017;

MPCobjSC=mpc(PlantSC,Ts2);

%Scale Factors
MPCobjSC.MV(1).ScaleFactor=pi/2;           %Steering Angle front in rad
MPCobjSC.MV(2).ScaleFactor=pi/2;           %Steering Angle rear in rad
MPCobjSC.OV(1).ScaleFactor=10;             %for vy in m/s
MPCobjSC.OV(2).ScaleFactor=0.6981317;     %for r in rad/s

%Prediction and Control Horizon
MPCobjSC.PredictionHorizon=10;
MPCobjSC.ControlHorizon=2;

%Constraints
MPCobjSC.MV(1).Min=-((45*pi)/180);         %Front steering angle lower bound
MPCobjSC.MV(1).Max=((45*pi)/180);         %Front steering angle upper bound
MPCobjSC.MV(2).Min=-((45*pi)/180);         %Rear steering angle lower bound
MPCobjSC.MV(2).Max=((45*pi)/180);         %Rear steering angle upper bound

MPCobjSC.MV(1).RateMaxECR=pi/30;          %Front steering angle rate
MPCobjSC.MV(2).RateMaxECR=pi/30;          %Rear steering angle rate

%Weights
MPCobjSC.Weights.OV(1)=20;
MPCobjSC.Weights.OV(2)=5;

ct_plant2=MPCobjSC.Model.Plant;
dt_plant2=c2d(ct_plant2,Ts2);

```

```

MPCobjSC.Model.Plant=dt_plant2;

review(MPCobjSC);

%% MPC for Torque Vectoring Control

Bf=1.4;
Br=1.4;
df=1*10^-1;      %Initial front steering angle for the vehicl model
dr=1*10^-1;      %Initial rear steering angle for the vehicl model

Iw=1.5;
rw=0.2521;
kxf=Caf*1.5;
kxr=Car*1.5;

O1=(-2*Caf-2*Car)/(m*vx); O2=((2*Car*Lr-2*Caf*Lf)/(m*vx))-vx; O3=0; O4=0;
O5=0; O6=0;
O7=(2*Car*Lr-2*Caf*Lf)/(Iz*vx); O8=(-2*Caf*Lf*Lf-2*Car*Lr*Lr)/(Iz*vx);
O9=(-kxf*Bf)/Iz; O10=(kxf*Bf)/Iz; O11=(-kxr*Br)/Iz; O12=(kxr*Br)/Iz;
O13=0; O14=0; O15=(-rw*rw*kxf)/(vx*Iw); O16=0; O17=0; O18=0;
O19=0; O20=0; O21=0; O22=(-rw*rw*kxf)/(vx*Iw); O23=0; O24=0;
O25=0; O26=0; O27=0; O28=0; O29=(-rw*rw*kxr)/(vx*Iw); O30=0;
O31=0; O32=0; O33=0; O34=0; O35=0; O36=(-rw*rw*kxr)/(vx*Iw);

P1=(2*Caf*df)/m; P2=(2*Car*dr)/m; P3=0; P4=0; P5=0; P6=0;
P7=(2*Caf*Lf*df)/Iz; P8=(-2*Car*Lr*dr)/Iz; P9=0; P10=0; P11=0; P12=0;
P13=0; P14=0; P15=-rw/(vx*Iw); P16=0; P17=0; P18=0;
P19=0; P20=0; P21=0; P22=-rw/(vx*Iw); P23=0; P24=0;
P25=0; P26=0; P27=0; P28=0; P29=-rw/(vx*Iw); P30=0;
P31=0; P32=0; P33=0; P34=0; P35=0; P36=-rw/(vx*Iw);

O=[O1 O2 O3 O4 O5 O6; O7 O8 O9 O10 O11 O12; O13 O14 O15 O16 O17 O18;
   O19 O20 O21 O22 O23 O24; O25 O26 O27 O28 O29 O30; O31 O32 O33 O34 O35
   O36];
P=[P1 P2 P3 P4 P5 P6; P7 P8 P9 P10 P11 P12; P13 P14 P15 P16 P17 P18;
   P19 P20 P21 P22 P23 P24; P25 P26 P27 P28 P29 P30; P31 P32 P33 P34 P35
   P36];
Q=[1 0 0 0 0 0; 0 1 0 0 0 0];
R=[0 0 0 0 0 0; 0 0 0 0 0 0];
PlantTV=ss(O,P,Q,R);

PlantTV.InputName={'n1','n2','Tfl','Tfr','Trl','Trr'};
PlantTV.StateName={'vy','r','sfl','sfr','srl','srr'};
PlantTV.OutputName={'vy','r'};
PlantTV.InputGroup.MV=6;
PlantTV.OutputGroup.MO=2;

TsTV=0.017;

MPCobjTV=mpc(PlantTV,TsTV);

MPCobjTV.MV(1).ScaleFactor=1.57;      %Steering Angle front in rad
MPCobjTV.MV(2).ScaleFactor=1.57;      %Steering Angle rear in rad
MPCobjTV.MV(3).ScaleFactor=1000;      %Torque Front Left in Nm
MPCobjTV.MV(4).ScaleFactor=1000;      %Torque Front Right in Nm
MPCobjTV.MV(5).ScaleFactor=1000;      %Torque Rear Left in Nm
MPCobjTV.MV(6).ScaleFactor=1000;      %Torque Rear Right in Nm

MPCobjTV.OV(1).ScaleFactor=2;          %for vy in m/s

```

```

MPCobjTV.OV(2).ScaleFactor=0.4;           %for r in rad/s

%Prediction and Control Horizon
MPCobjTV.PredictionHorizon=10;
MPCobjTV.ControlHorizon=2;

%Constraints
MPCobjTV.MV(1).Min=1;
MPCobjTV.MV(1).Max=1;
MPCobjTV.MV(2).Min=1;
MPCobjTV.MV(2).Max=1;
% Torque Constraints for normal driving
MPCobjTV.MV(3).Min=-50;
MPCobjTV.MV(3).Max=50;
MPCobjTV.MV(4).Min=-50;
MPCobjTV.MV(4).Max=50;
MPCobjTV.MV(5).Min=-50;
MPCobjTV.MV(5).Max=50;
MPCobjTV.MV(6).Min=-50;
MPCobjTV.MV(6).Max=50;

MPCobjTV.MV(3).RateMaxECR=5;
MPCobjTV.MV(4).RateMaxECR=5;
MPCobjTV.MV(5).RateMaxECR=5;
MPCobjTV.MV(6).RateMaxECR=5;

%Weights
MPCobjTV.Weights.OV(1)=1;
MPCobjTV.Weights.OV(2)=20;
MPCobjTV.Weights.MVRate=[0.01 0.01 20 20 20 20];

ct_plant2=MPCobjTV.Model.Plant;
dt_plant2=c2d(ct_plant2,TsTV);
MPCobjTV.Model.Plant=dt_plant2;

review(MPCobjTV);

```