Optimal Control of a Multi-Actuated By-Wire Vehicle

S.L.H. Verhoeven

DCT 2008.112

Traineeship report

Coach: Dr. C. Manzie

Supervisor: Prof.dr.ir. M. Steinbuch

Technische Universiteit Eindhoven
Department of Mechanical Engineering
Dynamics and Control Technology Group

Eindhoven, October, 2008
Optimal Control of a Multi-Actuated By-Wire Vehicle
SANDER LEONARD HENDRIK VERHOEVEN

Traineeship report

The University of Melbourne
School of Engineering
Department of Mechanical Engineering
Parkville Victoria 3010
Australia

Technische Universiteit Eindhoven
Department of Mechanical Engineering
Dynamics and Control Technology Group
Contents

1 Introduction 5

2 Modern Electronic Stability Control 7
  2.1 Introduction .................................................. 7
  2.2 Notation and Definitions ....................................... 7
  2.3 Principal mode of function ..................................... 9
  2.4 Existing ESC systems .......................................... 10
  2.5 Conclusion .................................................... 12

3 Vehicle Modelling 13
  3.1 Introduction .................................................. 13
  3.2 Vehicle Size and Shape ....................................... 13
  3.3 Powertrain .................................................... 14
  3.4 Brake System .................................................. 14
  3.5 Steering System ............................................... 15
  3.6 Suspension ................................................... 16
  3.7 Tyres ........................................................ 18
  3.8 Model Verification ............................................. 18
  3.9 Conclusion .................................................... 19

4 Controller design 21
  4.1 Introduction .................................................. 21
  4.2 Electronic stability controller design ........................ 22
  4.3 Control allocation ............................................. 23
Chapter 1

Introduction

An extension of the widely applied anti-lock braking system (ABS) and the traction control system (TCS), is the electronic stability control (ESC) system. Whereas ABS prevents the wheels from slipping when braking, and TCS prevents the wheels from spinning when accelerating, ESC helps drivers to retain control of their vehicles during high-speed maneuvers or on slippery roads.

The first ESC system was co-developed by Mercedes-Benz and Robert Bosch GmbH between 1987 and 1992. In 1995 the product was first offered to the public by Volvo and Mercedes-Benz, and in 1996 it acquired world wide fame when a journalist of a car magazine rolled a Mercedes-Benz A-class (without ESC) in a moose test (swerving to avoid an obstacle). As a consequence Mercedes-Benz recalled and retrofitted 130,000 A-class cars with ESC. After that the number of cars with ESC rose and at the moment most car manufacturers equip their cars standard with ESC - especially SUVs - or offer it as an option to the customer [Tec06].

A potential problem for increasing customer awareness is that car manufacturers use different names for the ESC system like Electronic Stability Program (Volkswagen Group, Mercedes-Benz, Holden), Dynamic Stability Control (BMW, Mazda), and StabiliTrak (General Motors).

Numerous studies around the world have been conducted to investigate the effectiveness of ESC in helping the driver to maintain control over his vehicle and thereby reducing the number and severity of crashes. According to an investigation conducted by the University of Cologne 4,000 lives can be saved and 100,000 accidents avoided if all European cars have ESC. A similar study done by the Insurance Institute for Highway Safety for North America [HIS06] shows that ESC could prevent nearly one-third (10,000) of all fatal crashes and reduce rollover risk by 80 percent for both single- and multiple-vehicle crashes. Needless to say ESC is one of the most important advances in automotive safety since the introduction of the seat belt.

Most ESC systems operate by braking individual wheels and/or reduce excess engine power if the driver is about to lose control. The focus of this report is to look at wheel steer as an additional actuator to assist the brakes in controlling a full by-wire vehicle. Including this extra actuator is realistic next step since one of the trends in automotive technology is to make the vehicle more by-wire, meaning that traditional mechanical and hydraulic control systems are being replaced with electronic control systems using electromechanical actuators.
and human-machine interfaces such as pedal and steering feel emulators.

Outline of the report

In order to design an electronic stability controller for a full by-wire vehicle the first thing that has to be done is to investigate the existing stability control systems. This is done in Chapter 2 by studying the original ESC system developed by Robert Bosch GmbH and a similar system modelled by the “Institut für Kraftfahrwesen” in Aachen, which is referred to as IKA in the remainder of the report. Chapter 3 then discusses the modelling of a vehicle to test the ESC that has to be designed. The test vehicle modelled in this report is the BMW 330xi of the year 2000 and the vehicle modelling software package that is used is Carsim 7. An accurate description of the model is desirable because further research in the field may be done with different vehicle modelling software. Chapters 4 and 5 describe the design and implementation of the stability controller. Because of limited time, however, it is not possible to design and model every part of the controller and certain assumptions have to be made. Chapter 6 contains some concluding remarks and recommendations for further research.
Chapter 2

Modern Electronic Stability Control

2.1 Introduction

The goal of this chapter is to familiarise the reader with the basic principle of electronic stability control and to have a closer look at some existing ESC systems. The next section first introduces the notation and sign convention that is used throughout the report as well as some common definitions and terms in the field of vehicle dynamics. After that the principle mode of function of vehicle stability controllers is explained and Section 2.4 then discusses two existing controllers and briefly touches upon the current state of research in the field.

The first design which is looked upon is the design from the IKA [CIT02]. They developed a simple version of an electronic stability controller which is based on the original design from Bosch. The second design that is discussed is the original controller developed by Bosch [ZEP95].

2.2 Notation and Definitions

Throughout the report the right-handed axis orientation given by ISO 8855 is used\(^1\), which has \(X\) pointing forward, \(Z\) pointing up, and \(Y\) pointing to the left of the vehicle. Other common terms are:

- **Longitudinal** - \(X\) component of force or translational motion vector,
- **Lateral** - \(Y\) component of force or translational motion vector,
- **Vertical** - \(Z\) component of force or translational motion vector,
- **Roll** - \(X\) component of moment or rotational motion vector,
- **Pitch** - \(Y\) component of moment or rotational motion vector,
- **Yaw** - \(Z\) component of moment or rotational motion vector.

Figure 2.1 shows the bicycle model (or single track vehicle model) of a car. As the name already suggests, the vehicle is modelled as a bike, consisting of only a longitudinal axis and two wheels. This 2-DOF model is not capable of describing the vehicle pitch and roll, but is

\(^1\)This is the same norm as used in Carsim [Say07]
Figure 2.1: Bicycle model of a vehicle.

very useful for analysing the cornering behavior [Pac02].

In Figure 2.1 the vehicle side slip angle is denoted by $\beta$ and is defined as the angle between 
the vehicle centre-line (or the vehicle longitudinal axis) and the direction of travel and can 
be approximated by

$$\beta = \tan^{-1}\left(\frac{-v_y}{v_x}\right) \approx \frac{-v_y}{v_x} \approx \frac{-v_y}{V},$$

(2.1)

for small vehicle side slip angles.$^2$ The wheel side slip angles, $\alpha_f$ and $\alpha_r$ are defined in a 
similar way. In the remainder of the report these side slip angles are simply called wheel slip 
angles, which is not to be confused with the longitudinal tyre slip ratio $\kappa$.

The lateral tyre forces $F_{y,f}$ and $F_{y,r}$ needed to manoeuver through a corner are generated 
by the tyres and for small wheel slip angles $\alpha$ and low longitudinal slip ratio $\kappa$ this can be 
approximated by

$$F_y = C\alpha,$$

(2.2)

with $C$ the cornering stiffness. In Chapter 3 it is shown that the behavior becomes strongly 
non-linear for large wheel slip angles, and that the cornering stiffness also depends on the 
vertical tyre load.

$^2$This is different from the definition used in Carsim [Gil07]
When a driver approaches a corner he tries to steer through the corner by turning the steering wheel. As a result the front wheels turn with angle $\delta$, leading to wheel slip angles on the front wheels. At this time the rear wheels do not know what is going on at the front, and as the lateral tyre forces are being built up in the front wheels the vehicle starts turning, which induces slip angles at the rear wheels as well.

During cornering the tyres need to develop lateral forces to keep the vehicle on a fixed radius $R$ at a certain forward velocity. If the forward velocity is increased while the cornering radius $R$ needs to be maintained the lateral tyre forces have to increase. If this increase in lateral tyre forces can be accomplished without modifying the steering angle it is called neutral steer. The increase in lateral tyre forces in then caused by a more ‘nose inwards’ orientation of the vehicle which increases both $\alpha_f$ and $\alpha_r$. If the increase in lateral force for the front tyre is too small, the driver has to increase the steering angle $\delta$ which is called understeer. If the increase in lateral force for the front tyre is too big, the driver has to decrease the steering angle $\delta$ which is called oversteer.

### 2.3 Principal mode of function

In a severe understeer reaction the magnitude of the actual yaw velocity, $\dot{\psi}$, is lower than the desired yaw velocity, $\dot{\psi}_{des}$, which can be derived from the vehicle speed and steering wheel angle. To increase the yaw velocity the electronic stability controller brakes the inner rear tyre, which creates a yaw moment/acceleration in the correct direction, see Figure 2.2.

If there is an oversteer reaction, i.e. the rear threatens to swerve, there is a fast increase in yaw velocity. The actual yaw velocity is higher than the desired yaw velocity and the stability controller initiates a brake force on the front wheel at the outside of the bend. In this way a yaw moment is created that counteracts the turning motion, see Figure 2.2.

In order to generate a desired yaw moment, one side of the vehicle could also be braked on both wheels. However, a brake intervention on the axle which almost or already reached its adhesion level is normally avoided.

Alongside the “yaw velocity controller” the vehicle side slip angle is also controlled. The necessity of this is made clear in Figure 2.3. Here the bottom curve (1) shows the path on a normal high friction road (high $\mu$ surface). The friction between tyre and road is sufficient for the lateral tyre forces to be transmitted and the vehicle is able to follow the nominal trajectory.

If the road surface friction coefficient is too low to achieve the nominal lateral acceleration needed to follow the path, a vehicle without yaw velocity control will understeer and swerve out of the curve (2). In a vehicle with yaw velocity control, but without vehicle side slip control, this situation would be evaluated as severe understeering and the controller would brake the rear wheel on the inside of the bend. This leads to a correct yaw velocity, but strong - undesired - vehicle side slip (3).

A stable vehicle movement therefore needs both a yaw velocity controller and a side slip limiter (4).
2.4 Existing ESC systems

In this section the electronic stability controller used by the IKA and the original design by Bosch are briefly discussed. The main goal of the IKA was to develop a method to check and monitor the status of the stability controller during driving. The controller they developed is based on the original ESC system developed by Bosch, but is simplified at some points since the focus was not on designing a high performance controller.

Figure 2.4 shows a simple version of the controller that was implemented. Modelling of sensor noise, parameter estimators, traction control, and plausibility checks is omitted here and the
ABS controller is assumed to be within the vehicle model. Under normal driving conditions the driver is able to follow the reference trajectory $x_{ref}$ without problems. However, when approaching the limit of the vehicle driving performance (e.g. low coefficient of friction, evasive manoeuvre at high speed), the driver is not able to control the vehicle any more (too much control effort). Under these situations the stability controller is there to assist the driver by individually controlling the brakes.

![Figure 2.4: Simplified version of the ESC system implemented in [CIT02].](image)

The basic working principle of the ESC system is as follows. The vehicle speed and the wheel steer angle are used to determine the desired yaw velocity $\dot{\psi}_{des}$ under normal tyre-road friction situations ($\mu = 1$). With the desired yaw velocity known the difference with the actual yaw velocity can easily be calculated using a simple substraction. The attained difference in yaw velocity and the actual yaw velocity are fed to the yaw velocity controller, which determines an appropriate braking action. The vehicle side slip controller works in parallel with the yaw velocity controller. Both controllers determine individual wheel brake torques at all time instants and a switch determines which brake torques are past on the the vehicle.

The yaw velocity controller and side slip angle limiter are implemented by applying fixed brake levels if certain thresholds are exceeded. For a “medium understeer” driving situation this may be implemented as

\[ \text{if} \left( \dot{\psi}_{des} - \dot{\psi} \right) > \text{understeer\_medium}, \text{then: Right rear brake = max brake value}. \]

The structure of the original controller designed by Bosch is different in a couple of ways. Without going into too much detail only the main differences are stated below [ZEP95].

- Besides determining the desired yaw velocity, $\dot{\psi}_{des}$, a desired vehicle side slip angle is also determined, $\beta_{des}$, by a “nominal value calculator”. In determining these variables the friction of the road $\mu$ is also estimated, which therefore effectively incorporates a side slip angle limiter.\(^\text{3}\)

---

\(^{3}\)In order to achieve this more variables than depicted in Figure 2.4 are needed.
Both the desired yaw velocity and vehicle side slip angle are compared with their actual (measured or estimated) values and then passed on to a state space feedback controller, which derives a nominal yaw moment.

From this nominal yaw moment and by using a vehicle model together with the actual values of longitudinal tyre slip, $\kappa$ and tyre slip angles $\alpha$, the required change in nominal slip value of each tyre is computed: $\Delta \kappa$. This change in tyre slip is then passed to a brake slip controller (ABS), which eventually leads to a change in vehicle yaw acceleration.

Since stability controllers play a major role in saving lives a lot of research has been conducted since the development of the first ESC system by Bosch. A good example is [And07]. Here two vehicle control strategies have been developed to deal with a large amount of actuators under the realistic assumptions that future vehicle development will lead to an increased amount of available actuators and that the onboard computational power will continue to increase. Although the results seem promising, the controllers were not tested in conjunction with a realistic vehicle model. This report focuses on including only steering as an additional actuator and on testing the designed controller on a realistic vehicle model. The control problem is also solved using a different technique, but the basic idea is the same.

2.5 Conclusion

In this chapter the basic principle of an electronic stability controller was introduced, together with two means of implementation. The first one being a relative simple method designed by the “Institut für Kraftfahrwesen”. This institute needed a simple implementation method that gave sufficient accurate results for their research goal, being the design of a method to monitor the status of the stability controller during driving. The second model was a simplified version of the stability controller developed by Bosch. These two models are used as a reference in Chapter 4 to build a new type of stability controller which also uses steering - besides braking - as a mean to control vehicle handling. Since it is desired to test the controller on a realistic car model the modelling of the test vehicle is described in the next chapter.
Chapter 3

Vehicle Modelling

3.1 Introduction

In order to test the ESC system that is designed is Chapter 4 a vehicle model is needed. This chapter describes the vehicle being modelled and the creation of a vehicle model using Carsim 7. As was pointed out in Chapter 1, the vehicle modelled in this report is the BMW 330xi of the year 2000. The reason for modelling this vehicle is that a lot of literature is readily available that describes the vehicle parameters and the modelling process in sufficient detail.

There are three reports used to extract vehicle parameters. The first is a report published by the IKA, which describes the modelling of two vehicles, including the BMW 330xi. In that report, however, Simpack is used as vehicle modelling software and therefore not all parameters needed in Carsim are known. The second source used is a Masters thesis done by S. Oliver at the University of Melbourne [Oli07]. In here a BMW 3.28i, which is a very similar vehicle, is modelled and the modelling has been done in Carsim 5.16b. The third reference is a report written by students of the University of Melbourne as part of their final year project [HD07]. The goal of the project was to remodel the BMW 330xi from the IKA report in Carsim 5.16b and implement a simple vehicle stability controller using Matlab/Simulink.

In the following every Carsim 7 element of the vehicle model is described. These elements are parameters related to the vehicle size and shape, powertrain, brake system, steering system, tyres, and suspension. Finally a short model validation is performed. For some parameters a Carsim best guess value is used, meaning that the value is based upon similar cars pre-modelled in Carsim. The cars in Carsim used for extracting these parameters are a C-class Hatchback, like the Audi A3, and a D-class Sedan, like the BMW 5 series.

3.2 Vehicle Size and Shape

In [CIT02] the location of the centre of gravity and the moments of inertia are experimentally determined for the whole vehicle, including two persons and measurement equipment. Results are shown in Table 3.1
Table 3.1: Vehicle size and shape parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wheel base</td>
<td>2725 mm</td>
</tr>
<tr>
<td>Track width front</td>
<td>1471 mm</td>
</tr>
<tr>
<td>Track width rear</td>
<td>1478 mm</td>
</tr>
<tr>
<td>Dynamic tyre radius</td>
<td>318.4 mm</td>
</tr>
<tr>
<td>Total mass</td>
<td>1725 kg</td>
</tr>
<tr>
<td>Distance front axle to cog</td>
<td>1365 mm</td>
</tr>
<tr>
<td>Unsprung mass (Front axle / rear axle)</td>
<td>80 kg / 80 kg</td>
</tr>
<tr>
<td>Unsprung mass inertia (Front axle / rear axle)</td>
<td>0.9 kgm² / 0.9 kgm²</td>
</tr>
<tr>
<td>Cog height</td>
<td>493 mm</td>
</tr>
<tr>
<td>Moment of inertia around x-axis (longitudinal)</td>
<td>510 kgm²</td>
</tr>
<tr>
<td>Moment of inertia around y-axis (lateral)</td>
<td>2280 kgm²</td>
</tr>
<tr>
<td>Moment of inertia around z-axis (vertical)</td>
<td>2730 kgm²</td>
</tr>
</tbody>
</table>

The vehicle size and mass parameters for the whole vehicle can easily be entered in Carsim 7. However, when using an older version of Carsim this is not possible, and the parameters need to be converted to parameters for the sprung mass only. This is not a very difficult task (only using Steiner’s law already gives good results), but care must be taken while doing it since errors in mass and inertias greatly influence the handling of the vehicle.

3.3 Powertrain

As the BMW 330xi is more a rear-dominated vehicle despite its four wheel drive, it is modelled as a rear-wheel driven vehicle. Since the powertrain is - at least at this stage - not of significant importance to the vehicle handling, the powertrain is modelled using standard Carsim components.

The vehicle model is therefore equipped with a standard 150 kW engine that is pre-modelled in Carsim7 and has an accompanying torque converter, standard 6-speed transmission and standard differential with a 4.1 ratio. The reason for not applying the drive torque directly to the wheels is that it would cancel out the effect of the rear differential, which is not desired.

3.4 Brake System

The braking system is modelled according to [CIT02]. In that report the brake power at the wheels is measured as a function of the brake pressure. It is not clear, however, if the pressure is measured in the individual brake cylinders or in the main brake cylinder. Since the brake force distribution is derived from the resulting graphs, it is more likely that the pressure is measured in the main cylinder. The resulting relations between brake torque and brake pressure are then 237 Nm/MPa for the front axle and 117 Nm/MPa for the rear axle, and both relations are independent of speed. This is considerably higher than in [HD07] where the brake torque/brake pressure ratios are only 97.8 Nm/MPa and 48.6 Nm/MPa for the front and rear axle, respectively.
The fluid pressure proportioning is modelled as “unity gain” for all four tyres because it is assumed that the pressure was measured in the main cylinder and not in the individual brake cylinders. Therefore, the effect of brake force distribution is already included in the brake torque vs. pressure relationship and is assumed constant. This is a big difference with [HD07] because there the fluid proportioning was modelled as a unity gain for the front axle and a constant 40% (of the front axle) for the rear axle.

Carsim 7 offers the option to utilise a build-in ABS controller. It is not necessary to use it at this stage, so it is effectively disabled by setting the ABS cut-off speed to 200 km/h.

### 3.5 Steering System

Carsim 7 allows for much more accurate modelling of the steering mechanisms in a car. It includes modelling of the power steering, hysteresis effects of the steering column, a variable steering ratio and much more. There is, however, only limited information available about these features and therefore the simple steering option is used, which allows for backward compatibility with older Carsim versions. The parameters used in this report and that were used in [HD07] are depicted in Table 3.2. The main differences are the absence of rear wheel steering and the 50% Ackermann steering, which is common in most cars and small trucks.

<table>
<thead>
<tr>
<th>Table 3.2: Steering system parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nominal steering gear ratio</strong></td>
</tr>
<tr>
<td><strong>Rear steering: steer gain vs. speed</strong></td>
</tr>
<tr>
<td><strong>Road wheel steer vs. Gear-down input steer</strong> (front axle)</td>
</tr>
<tr>
<td><strong>Road wheel steer vs. Gear-down input steer</strong> (rear axle)</td>
</tr>
<tr>
<td><strong>Front compliance</strong></td>
</tr>
<tr>
<td><strong>Rear compliance</strong></td>
</tr>
<tr>
<td><strong>Kingpin geometry</strong></td>
</tr>
<tr>
<td>Steering wheel torque / total kingpin moment</td>
</tr>
<tr>
<td>Lateral offset at centre (front axle / rear axle)</td>
</tr>
<tr>
<td>Kingpin inclination (front axle / rear axle)</td>
</tr>
<tr>
<td>X-coordinate of kingpin at centre (front axle / rear axle)</td>
</tr>
<tr>
<td>Caster angle (front axle / rear axle)</td>
</tr>
</tbody>
</table>

\[4\text{Value is based on best guess by comparing the vehicle with a C-class Hatchback and a D-class sedan.}\]

\[5\text{All rear axle values are set to zero, because the rear wheels cannot be steered.}\]
3.6 Suspension

Just as with the modelling of the steering system the suspension modelling options are also extended in Carsim 7. The major changes are, however, in the GUI. Backward capability is again possible, but it is not used in this report. Carsim 7 splits the modelling up in two parts: suspension kinematics and suspension compliance.

Suspension kinematics

Carsim 7 has several types of suspensions that are already modelled, like solid axle, tri-link, 5-link and MacPherson strut suspension. Using the internet it was discovered that the BMW 330xi has independent MacPherson strut suspension for its front wheels and 5-link suspension for both its rear wheels. Therefore the kinematics for the front and rear axle are modelled using the models readily available in Carsim 7, meaning standard relations for “wheel dive moment due to jounce”, “wheel roll movement due to jounce”, and “toe (steer) due to jounce”. Furthermore, the jounce (vertical wheel movement) is defined from spring data. Additional parameters are depicted in Table 3.3.

<table>
<thead>
<tr>
<th>Unsprung mass</th>
<th>80 kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spin inertia for each wheel</td>
<td>0.9 kgm$^2$</td>
</tr>
<tr>
<td>Fraction steered of unsprung mass - front axle</td>
<td>0.8</td>
</tr>
<tr>
<td>Fraction steered of unsprung mass - rear axle</td>
<td>0.1</td>
</tr>
<tr>
<td>Height wheel centre</td>
<td>Dynamic tyre radius</td>
</tr>
<tr>
<td>Static alignment settings</td>
<td></td>
</tr>
<tr>
<td>Camber - left</td>
<td>0 deg</td>
</tr>
<tr>
<td>Camber - right</td>
<td>0 deg</td>
</tr>
<tr>
<td>Toe - left</td>
<td>0 deg</td>
</tr>
<tr>
<td>Toe - right</td>
<td>0 deg</td>
</tr>
</tbody>
</table>

Suspension compliance

In the suspension compliance menu the springs, shock absorbers and compliance effects can be specified. The compliance coefficients effects relate steer, camber, lateral displacement and longitudinal displacement of the wheel to tyre forces and moments. The coefficients used in this report to describe these effects are the same as for the Carsim C-class Hatchback and are displayed in Table 3.4. These coefficients are also very similar for the D-class sedan. The only exception is the Toe / $F_x$ coefficient for the front axle, which is zero for the D-class sedan.

The ratios between spring (and shock absorber) compression and suspension jounce are set to one, meaning that one unit of vertical wheel displacement is equal to one unit of spring (and damper) displacement. This setting is not true for real vehicles because springs and shock absorbers are always installed under a certain angle. In the IKA report, however, the spring and damper forces are measured as a function of vertical wheel displacement and therefore this modelling step is allowed.
Table 3.4: Suspension compliance coefficients

<table>
<thead>
<tr>
<th></th>
<th>Front axle</th>
<th>Rear axle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toe / $F_x$</td>
<td>$0.43 \cdot 10^{-3}$ deg/N</td>
<td>0 deg/N</td>
</tr>
<tr>
<td>Steer / $F_y$</td>
<td>$-0.27 \cdot 10^{-3}$ deg/N</td>
<td>$-0.10 \cdot 10^{-4}$ deg/N</td>
</tr>
<tr>
<td>Steer / $M_z$</td>
<td>$0.38 \cdot 10^{-2}$ deg/Nm</td>
<td>$0.19 \cdot 10^{-2}$ deg/Nm</td>
</tr>
<tr>
<td>Camber / $F_x$</td>
<td>0 deg/N</td>
<td>0 deg/N</td>
</tr>
<tr>
<td>Inclination / $F_y$</td>
<td>0 deg/N</td>
<td>$0.29 \cdot 10^{-3}$ deg/N</td>
</tr>
<tr>
<td>Inclination / $M_z$</td>
<td>0 deg/Nm</td>
<td>0 deg/Nm</td>
</tr>
<tr>
<td>Longitudinal displacement / $F_x$</td>
<td>$5.0 \cdot 10^{-5}$ mm/N</td>
<td>$7.0 \cdot 10^{-5}$ mm/N</td>
</tr>
<tr>
<td>Lateral displacement / $F_y$</td>
<td>$4.1 \cdot 10^{-5}$ mm/N</td>
<td>$3.8 \cdot 10^{-5}$ mm/N</td>
</tr>
</tbody>
</table>

The spring characteristics are depicted in Figure 3.1. The springs are modelled as linear approximations of the spring characteristics given in the IKA report and bump stops are also included. The linear spring stiffness coefficients are set to 35 N/mm and 31 N/mm for the springs in the front and rear suspension, respectively. Furthermore, it is assumed that there is no spring deflection from the centre in the steady state situation. To accomplish this the spring characteristic of the front suspension was adjusted in vertical direction. Hence the vertical offset in the left hand side graph of Figure 3.1.

The shock absorbers are modelled using the same non-linear relations as given in the IKA report. Initially it is not clear if the authors classify a positive piston speed as compression or extension, but by comparing it with other shock absorber characteristics already available in Carsim, it is more likely that a positive piston speed is used to describe the extension phase$^6$. In Figure 3.2 the damper characteristics are depicted and compared with the damper characteristic of a C-class hatchback and a D-class sedan. It is clear that there is a big difference between the characteristics, but this has partly to do with the compression/jounce ratio mentioned earlier. However, even after correction the difference is still significant.

$^6$This is different from [HD07], where the opposite is assumed
The auxiliary roll moment provides resistance when the vehicle is rolling, e.g. when the vehicle is cornering. In the IKA report the total effective spring force due to the normal spring, which is modelled earlier, and the stabilizer bar is measured. For the sake of simplicity, however, the default Carsim values are used, which are 384 Nm/deg and 344 Nm/deg for the front and rear axle, respectively. The auxiliary roll damping is set to zero, which is also a Carsim default.

3.7 Tyres

The tyres of the vehicle are its connection to the ground and all forces between the vehicle and the ground are transferred via the tyres. The tyres are therefore of crucial importance to the vehicle behaviour, especially while cornering, and accurate modelling is essential. In contrast to the references mentioned in the introduction the standard 205/50 R17 tyres are modelled using a pre-modelled tyre model available in Carsim 7. Figure 3.3 shows the longitudinal tyre force as a function of the longitudinal tyre slip ratio and the lateral tyre force as a function of the wheel side slip angle.

3.8 Model Verification

Because the main goal this study is to investigate a new type of vehicle stability controller, the model does not have to be perfect. It is, however, desired that the model behaves like a real vehicle in normal driving situations. A small test is performed to check if the vehicle behaves realistically in a very common - and for this study relevant - situation. The situation is a double lane change at 150 km/h using the internal driver model of Carsim. The test is relevant because a double lane change at high speeds can cause too much understeer, which the ESC systems tries to diminish.

The expected result is cornering behaviour that is a bit worse (more swerving) than with a normal C-class hatchback, but better than or as good as an unloaded rear wheel driven
Figure 3.3: Longitudinal tyre force as a function of longitudinal wheel slip (left) and lateral tyre force as a function of the wheel slip angle (right).

D-class sedan. Results are shown in Figure 3.4 and are good enough to allow the model to be used for testing the stability controller.

Figure 3.4: Vehicle side slip angle (left) and yaw velocity (right) during a double lane change with an offset of 3.5 m at 150 km/h.

3.9 Conclusion

In this chapter the modelling of a BMW 330xi test vehicle was described. This model is used in Chapters 4 and 5 to design and test the electronic stability control system that is designed in the next chapter. Since the goal of the study is to investigate a new type of ESC system, no extensive model validation has been performed. The model was, however, validated by comparing its response with similar pre-modelled Carsim vehicles.
Chapter 4

Controller design

4.1 Introduction

The chapter describes the use of model predictive control (MPC) theory to design an electronic
stability controller that utilises both steering and braking. The next section explains the
control approach that is used and gives an overview of how the controller is implemented.
Section 4.3 explains how the actual control allocation is formulated and how it can be solved.
After that the theory of Section 4.2 and 4.3 is used to develop an ESC system suitable for the
BMW 330xi and finally simulations are performed for understeer and oversteer situations.

An important task of the electronic stability controller is to distribute control power among
redundant control effectors, under a set of constraints. In this first controller design some
assumptions are made because the focus is mainly on the control allocation part and not on
how to interpret the driver’s commands (and estimation of environmental variables) to come
up with a desired yaw velocity and vehicle side slip angle. Some information about that part
of the controller design and the overall control structure can be found in [ZEP95].

The assumptions made for this chapter are

1. The desired yaw velocity is prescribed. Effectively this means that the driver’s com-
mands are already interpreted, and that there is no additional direct driver influence
on the vehicle, since it is a full by-wire vehicle.

2. The road has a high coefficient of friction (high $\mu$ surface) so there is no need for a
vehicle side slip limiter.

3. Initially the actuator dynamics are infinitely fast and can therefore be neglected. In
Chapter 5 the actuator dynamics are not neglected, leading to a slightly different ap-
proach.

4. In the vehicle model used for the control allocation the relation between actuator output
and vehicle input (yaw acceleration/moment) is assumed to be constant and linear.

The control problem left to be solved is then stated as
“distribute control power under a set of redundant effectors to let the vehicle track a prescribed yaw velocity”.

The problem is solved in a similar way as is done in literature [VSB07], [LSY+04], [LSY+05], and [LSY+07]. In the latter three sources the authors use MPC for their control allocation strategy for re-entry vehicles (RVs), i.e. vehicles that have to re-enter the earth’s atmosphere.

4.2 Electronic stability controller design

For an overactuated system one can often identify a signal (often a force or moment) that characterises the overall effect of many actuators, which acts as “virtual control”. For the vehicle stability controller one can think about a yaw acceleration or yaw moment, which is similar to aircraft control [LSY+04], [LSY+05], [LSY+07], where the three moments around the centre of gravity are used. The introduction of the virtual control decomposes the control system into two parts leading to a modular approach, see Figure 4.1. Here, an outer loop controller is used to determine the desired virtual input, \( \dot{\psi}_{\text{des}} \), that has to be tracked by the inner loop controller. The inner loop controller then determines which actuators to use (control allocation) such that the vehicle eventually tracks the desired yaw velocity, \( \dot{\psi}_{\text{des}} \). The “Dynamic Inversion” block produces a correction on the desired yaw acceleration based on the vehicle model, which is further explained in the rest of this section. A more complete block scheme of the total stability controller that also incorporates the side slip limiter and an interpreter for the driver commands can be found in [ZEP95].

\[
\begin{align*}
\dot{\omega} &= f(\omega, \theta) + g(u, \theta), \\
(4.1)
\end{align*}
\]

with \( \omega \in \mathbb{R}^3 \) the angular velocity vector and \( \theta \in \mathbb{R}^p \) a vector containing measurable or estimable parameters, like tyre slip angle. The vector \( u \in \mathbb{R}^n \) contains the \( n \) (\( n = 5 \)) control actuator signals: \( u = [\delta \ T_{b,lf} \ T_{b,lr} \ T_{b,rf} \ T_{b,rr}]^T \). Here \( \delta \) is the steering wheel angle and \( T_{b,ij} \) is the brake moment with the first index denoting the left (\( l \)) or right (\( r \)) side of the vehicle and the second index denoting front (\( f \)) or rear (\( r \)) end of the vehicle.

For the design of the electronic stability controller the only relevant angular velocity at this stage is the yaw velocity and therefore Equation (4.1) can be rewritten as
\[
\dot{\psi} = f(\dot{\psi}, \theta) + g(u, \theta). \tag{4.2}
\]

The term \(f(\dot{\psi}, \theta)\) accounts for accelerations due to the vehicle body, whereas the term \(g(u, \theta)\) represents accelerations influenced by the actuators. From here on the non-linear term \(g(u, \theta)\) is assumed to be a constant linear mapping of the form

\[
g(u, \theta) = Gu \tag{4.3}
\]

with \(G \in \mathbb{R}^{1 \times 5}\) a constant matrix that is derived in Section 4.4. The desired yaw acceleration that the inner loop controller - the control allocation algorithm - has to track can now be adjusted for the acceleration that is generated by the vehicle body using

\[
y_{\text{des}} = \ddot{\psi}_{\text{des}} - f(\dot{\psi}, \theta), \tag{4.4}
\]

which is done with the “Dynamic Inversion” block. The control allocation problem which has to be solved in the “MPC-CA” block is then restated to finding input commands \(u\) such that \(y_{\text{des}} = Gu\), with \(u\) subject to certain constraints.

### 4.3 Control allocation

Model predictive control is used to distribute control effort between different actuators. A typical model used for this is shown in Figure 4.2. As stated in the introduction it is assumed that the actuator dynamics are infinitely fast and therefore \(u_{\text{cmd}} = u_{\text{act}} = u\), with \(u_{\text{cmd}}\) being the input signal to the actuators and \(u_{\text{act}}\) is the real actuator output. The input signal \(y_{\text{des}}\) is given by Equation (4.4) and \(y\) is the actual achieved yaw acceleration (or moment) delivered by the actuators, which is assumed to be a linear constant function of the actuator output, like in Equation (4.3).

![Figure 4.2: Model used for control allocation.](image)

The control allocation problem then takes the following form

find \(u\) such that

\[
y_{\text{des}} = Gu, \quad \text{s.t.} \quad u_{\text{min}} \leq u \leq u_{\text{max}}. \tag{4.5}
\]

If a feasible solution exists, the available actuator redundancy may be employed to satisfy a sub objective of the form
\[
\min_u J_{sub} = \min_u |u - u_p|^T W_p [u - u_p],
\]

where \(W_p\) is a weighting matrix and \(u_p\) is a preferred control input chosen to meet additional requirements.

In order to solve the static control allocation problem, Equation (4.5) and (4.6) are combined in a 1-norm mixed optimisation problem

\[
\min_u J_d = \min_u \left\{ \|y_{des} - Gu\|_1 + \lambda \|W_p (u - u_p)\|_1 \right\},
\]

where \(\lambda\) is a scalar weight between tracking error and control effort. If \(W_p\) is assumed to be diagonal, the 1-norm in Equation (4.7) allows the problem to be cast into the following LP problem

\[
\min_x J_d = \min_x \begin{bmatrix} 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} 1 & \cdots & 1 \end{bmatrix} \begin{bmatrix} y_{des} \\ -y_{des} \\ \vdots \\ -y_{des} \\ u_p \\ -u_p \end{bmatrix} = m^T x,
\]

where \(w_p\) is a vector containing the diagonal components of \(W_p\) and \(u_s\) and \(u_\sigma\) are slack variables with the same dimension as \(u\). The constraints that have to be satisfied are then written in linear form \(Ax \leq b\) as

\[
\begin{bmatrix} \tilde{G} & -I_n & 0 \\
-\tilde{G} & -I_n & 0 \\
I_n & 0 & -I_n \\
-I_n & 0 & -I_n \\
A \\
\end{bmatrix} \begin{bmatrix} u \\ u_s \\ u_\sigma \end{bmatrix} \leq \begin{bmatrix} y_{des} \\ y_{des} \\ \vdots \\ y_{des} \\ u_p \\ -u_p \end{bmatrix}, \quad u_s \geq 0, \quad u_\sigma \geq 0, \quad u_{min} \leq u \leq u_{max},
\]

with \(\tilde{G}\) a column containing \(n\) components \(G\), \(A \in \mathbb{R}^{4n \times 3n}\), \(b \in \mathbb{R}^{4n}\), \(I_n\) a unity matrix of size \(n\) and \(0\) a square matrix of size \(n\) filled with zeros. The obtained LP problem can then be solved using standard algorithms.

Rate constraints on control effort of the form \(|\dot{u}| \leq \bar{u}\) can be included by approximating the derivative of \(u\) with a first order difference approximation

\[
\dot{u} \approx \frac{u(t) - u(t - T_s)}{T_s},
\]

where \(T_s\) is the sampling time.
The new upper and lower bounds on $u$ are then given by

$$u = \max \left\{ u_{\min}, u(t - T_s) - \bar{u} T_s \right\}, \quad (4.10)$$

and

$$\bar{u} = \min \left\{ u_{\max}, u(t - T_s) + \bar{u} T_s \right\}. \quad (4.11)$$

### 4.4 ESC system for the BMW 330xi

In this section the theory described in the previous two sections is applied to develop an ESC system for the BMW 330xi and implement the controller in Matlab/Simulink.

The first item needed is a model which describes the effects of the actuators on the yaw acceleration of the vehicle. To model the effects of steering the bicycle model in Figure 2.1 is used. It is thereby assumed that the all angles $\delta$, $\alpha_f$, $\alpha_r$ are small. The variation of the geometry may therefore be regarded as linear: $\sin(x) \approx x$ and $\cos(x) \approx 1$. It is also assumed that the longitudinal tyre forces $F_{x,ij}$, (drive and brake forces) are small compared to the lateral tyre forces $F_{y,ij}$ and the influence of $F_{x,ij}$ on $F_{y,ij}$ can therefore be neglected. Assuming small brake forces is equivalent to assuming small longitudinal slip ratios $\kappa$, and by looking at the combined side force and brake force characteristics, it can be verified that the assumption is valid and that $F_y$ solely depends on $\alpha$ [Pac02]. The rotational equation of motion then becomes

$$I_{zz}\ddot{\psi} = -b F_{y,lr} - b F_{y,rr} + a F_{y,lf} + a F_{y,rf}, \quad (4.12)$$

with $a$ and $b$ being the distances between the centre of gravity and the front and rear axle, respectively. Since it is assumed that the side slip angles are small Equation (4.12) can be rewritten to

$$I_{zz}\ddot{\psi} = -2b C_r \alpha_r + 2a C_f \alpha_f, \quad (4.13)$$

with $C_f$ and $C_r$ the cornering stiffness of the front and rear wheels, respectively, and

$$\alpha_r = -\frac{1}{v_x}(v_y - b \dot{\psi}), \quad (4.14)$$

$$\alpha_f = \delta - \frac{1}{v_x}(v_y + a \dot{\psi}). \quad (4.15)$$

Combining Equation (4.13), (4.14), and (4.15) leads to the following equation of motion
\[
I_{zz} \ddot{\psi} = \frac{2}{v_x} [(a C_f - b C_r)] v_y - \frac{2}{v_x} [(a^2 C_f + b^2 C_r)] \dot{\psi} + 2a C_f \delta. \tag{4.16}
\]

In a similar way the influence of the brakes on the yaw acceleration can be modelled. For this the bicycle model is not used, but it is assumed that brake forces work parallel to the vehicle longitudinal axis (skateboard model), which is valid under the same small angle approximation for \( \alpha \) and \( \delta \). The contribution of the brakes to the yaw acceleration of the vehicle is then given by

\[
I_{zz} \ddot{\psi} = \frac{t_f}{2} \frac{T_{b,lf}}{r_{dyn}} + \frac{t_r}{2} \frac{T_{b,lr}}{r_{dyn}} - \frac{t_f}{2} \frac{T_{b,rf}}{r_{dyn}} - \frac{t_r}{2} \frac{T_{b,rr}}{r_{dyn}}, \tag{4.17}
\]

with \( r_{dyn} \) the dynamic tyre radius, and \( t_f \) and \( t_r \) the track width at the front and rear axle, respectively.

The constant matrix \( G \) used in the control allocation algorithm can then be derived by combining Equation (4.16) and (4.17)

\[
G = \frac{1}{I_{zz}} \begin{bmatrix} 2 C_f a & \frac{t_f}{2} & \frac{t_r}{2} & -\frac{t_f}{2} & -\frac{t_r}{2} \\ \frac{1}{r_{dyn}} & \frac{1}{r_{dyn}} & \frac{1}{r_{dyn}} & \frac{1}{r_{dyn}} & \frac{1}{r_{dyn}} \end{bmatrix}. \tag{4.18}
\]

In order to solve the optimisation problem the vector \( u_p \) has to be specified as well. It is desirable to brake as little as possible, so the \( u_p(i) \) values for braking are set to zero. It also desirable to steer as smoothly as possible, but the absolute value of the steering angle is not important as long as it does not violate the constraints. The \( u_p \) value for steering is therefore set to be the previous steering input, so that changes in steering angle are penalised.

### 4.5 Simulation results

To test whether the designed controller works three simulations are performed. The first one is a simulation to determine whether the control allocation algorithm and dynamic inversion block work correctly. To test this the vehicle is simulated as a 2-DOF model using Equation (4.16) and an equation of motion for the lateral vehicle velocity. After that the 2-DOF model is replaced by the Carsim model of Chapter 3 and simulations for an understeer and oversteer situation are performed. To carry out these simulations a co-simulation environment is set up between Carsim 7 and Matlab/Simulink. Besides modelling the car and solving the equations of motions with the build-in Carsim solver (VS Solver), Carsim 7 also allows the equations of motion to be solved with other software, like Simulink. How the co-simulation environment is set up is described in detail in Appendix B. The necessary files needed to repeat the simulations and the equations of motion of the 2-DOF car model are also given there.
Understeer test

The first test performed is an understeer test. An important difference with the existing ESC systems is that the system developed here is designed for a full by-wire vehicle (so no direct driver input). Therefore, all the manoeuvring has to be done by the controller instead of assisting the driver’s input. It is preferred to let the vehicle track the desired yaw velocity by steering through the manoeuvre. When, however, steering alone is not sufficient, the brakes should assist. In case of an understeer situation the rear brake on the inner site of the corner should be activated first, for reasons explained earlier. The main difference with a regular ESC system is that there the brakes are used immediately if the difference between actual yaw velocity and desired yaw velocity is large enough [ZEP95].

The reference yaw velocity signal that is used in the first two tests is depicted in Figure 4.3 and imitates a sharp turn to the left. In order to smooth the reference signal the yaw jerk is prescribed as a first order signal, leading to a third order reference yaw velocity with an end value of 12 deg/s. The manoeuvre is performed at a speed of 144 km/h (or 40 m/s). The lateral acceleration accompanied by this manoeuvre is around 0.85 g, indicating that the vehicle dynamics with this manoeuvre is non-linear. Weights, constraints and outer loop controller parameters can be found in Table 4.1.

Simulation results for the vehicle modelled as a 2-DOF linear model and using the Carsim model are given in Figure 4.4 and 4.5, respectively. In these figures the resulting yaw acceleration, tracking error and actuator signals are shown.

Figure 4.4 shows that the tracking is - as expected - very good, indication that the control allocation algorithm and dynamic inversion block work correctly. Increasing the gain leads to even better tracking, but too high gain values lead to chattering behaviour, because the optimisation algorithm is run in discrete time. It is, however, impossible to get zero tracking error for the entire time span, since a proportional feedback controller always requires an error to determine a control action. The maximum wheel steer angle is set to 0.5 deg, which is lower than for the simulation with the Carsim model. The reason for the low bound on wheel steer angle is to force the control allocator to use the brakes. The control allocator first tries to steer the vehicle through the corner, but when it is not allowed to steer anymore, it
Table 4.1: Simulation parameters for the understeer tests

<table>
<thead>
<tr>
<th></th>
<th>2-DOF model</th>
<th>Carsim model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outer loop controller</td>
<td>P = 1, I = 0, D = 0</td>
<td>P = 1, I = 5, D = 0.04</td>
</tr>
<tr>
<td>Optimisation frequency</td>
<td>80 hz</td>
<td>80 hz</td>
</tr>
<tr>
<td>Relative weight on control $\lambda$</td>
<td>0.3</td>
<td>0.3</td>
</tr>
<tr>
<td>Weights on actuators, $w_k$</td>
<td>$[1 2 1 2 1]^T$</td>
<td>$[1 2 1 2 1]^T$</td>
</tr>
<tr>
<td>Maximum wheel steer angle</td>
<td>0.5 deg</td>
<td>2 deg</td>
</tr>
<tr>
<td>Maximum brake torque at front axle</td>
<td>1000 Nm</td>
<td>1000 Nm</td>
</tr>
<tr>
<td>Maximum brake torque at rear axle</td>
<td>900 Nm</td>
<td>900 Nm</td>
</tr>
<tr>
<td>Maximum steering angle rate</td>
<td>900 deg/s</td>
<td>900 deg/s</td>
</tr>
<tr>
<td>(Absolute value at steering wheel)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Maximum brake torque rate</td>
<td>200000 Nm/s</td>
<td>200000 Nm/s</td>
</tr>
<tr>
<td>(Absolute value)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

starts using its left rear brake and after that its left front brake. With a higher bound on maximum steer angle the 2-DOF vehicle is capable of “just” steering through the corner.

Figure 4.4: Simulation results for an understeer scenario with the 2-DOF vehicle model. Top left graph: desired and actual yaw velocity, top right graph: tracking error, bottom left graph: wheel steer angle, and bottom right graph: brake torques.
Figure 4.5: Simulation results for an understeer scenario with the Carsim vehicle model. In this scenario the lateral acceleration is approximately 0.85 g. Top left graph: desired and actual yaw velocity, top right graph: tracking error, bottom left graph: wheel steer angle, and bottom right graph: brake torques.

The results for the Carsim model look quite similar. Again, the controller first tries to steer through the reference trajectory, but still needs the left rear brake because of the constraint on the steering angle. Around $t = 2.5$ s there is a small dip in vehicle yaw velocity. This has to do with non-linearities that are not accounted for in vehicle model used for the control allocation. The effect of the non-linearities can be made more clear by looking at the simulation results for the Carsim model, but with a maximum wheel steer angle of 0.5 deg, see Figure 4.6. The required brake torque at the left rear wheel is 400 Nm at $t = 4$ s, which is approximately 200 Nm lower than for the 2-DOF model, see Figure 4.4. Apparently, either the effect of steering or braking is underestimated, or both. One reason for underestimating the effect of steering is that the vehicle model is not able to describe vehicle roll. When the vehicle starts turning the vertical load on the outer wheels of the corner increases, leading to a higher total lateral force than modelled.

Another important non-linearity is that the actual wheel steer is different from the applied steering angle. An applied steering wheel angle of 2 deg in steady state cornering ($t > 4$ s) leads to a wheel steer angle of 1.8 deg on the inner side of the corner and only 0.3 deg at the outer side of the corner. Reasons for this non-linearity are the compliance coefficients of
A big problem for the ESC system is the choice of proper constraints. In the simulation the maximum wheel steer angle is set to 2 deg, which makes braking of the left rear wheel necessary. If, however, this constraint is set to 3 deg, the controller is able to guide the vehicle through the path by only steering its front wheels. The issue here is that when a larger steering angle is allowed, the tyre side slip angles become larger, and the operating point of the tyre is even more in the non-linear region, see Figure 3.3. With an allowed steering angle of 3 deg the maximum tyre slip angle at the left front wheel is approximately 5.6 deg, whereas with a maximum steering angle of 2 deg it is five degrees. The difference is not spectacular, but it indicates the need to choose the constraints wisely.

Another way of dealing with non-linearities is to make the model parameter dependent (e.g. speed, surface, side slip angle), like in Equation (4.2). This, however, does not take away the need for situation dependent constraints. Furthermore, the constraints set on the brake moment are set very low such that there is only a small amount of longitudinal wheel slip ($\kappa \approx 0$). For a more realistic simulation the tyre slip ratio, $\kappa$, should be used as input to the ABS controller of the vehicle. Constraints on $\kappa$ can be set more apprehensible, but it also requires changing the vehicle model. In [And07] simulations are performed with a simple vehicle model, but with a more complex tyre model that is able to describe the combined effect of lateral and longitudinal tyre forces.

**Oversteer test**

In order to induce an oversteer situation a force of 8000 N is placed on both rear wheels in negative y-direction for a duration 0.2 s, starting at $t = 1$ s and driving at 144 km/h. At the same time the desired yaw velocity is kept zero. This scenario causes the rear end of the vehicle to be pulled out of the intended direction of motion. The simulation parameters are kept almost the same as in Table 4.1. The only changes are that the weights on braking the
front and rear axle are switched and the maximum absolute steering angle is set to 10 deg instead of 2 deg, allowing for brake tyre forces that are clearly not parallel to the longitudinal axis of the vehicle anymore. Results are given in Figure 4.7 and Figure 4.8.

![Figure 4.7: Tracking of yaw velocity for an oversteer scenario with the Carsim vehicle model.](image)

![Figure 4.8: Actuator outputs for an oversteer scenario with the Carsim vehicle model. Left graph: wheel steer angle, and right graph: brake torques.](image)

It can be seen that the controller tries to counteract the side force by steering and braking both wheels on the right side of the vehicle simultaneously and at maximum power. This is, however, not possible in one time step because of the rate constraints. Figure 4.7 clearly shows that the yaw velocity of the vehicle becomes negative for more than half a second. This is due to the integral action which basically forces the yaw error to zero, causing the vehicle to continue its path in the original direction. Without the integral action the yaw velocity would go to zero faster (no overshoot), but this also means that the vehicle would be driving in a different direction (different yaw angle). In a real application the yaw velocity set point would not be kept constant during such a manoeuvre, but a nominal value estimator would come up with new a setpoint every $T_s$ seconds, which is fed to the outer loop controller [ZEP95].
4.6 Conclusion

In this chapter the use of model predictive control theory for designing a control allocation algorithm to use in an electronic stability controller for a vehicle was described. After introducing the theory from literature [VSB07], [LSY+04], [LSY+05], and [LSY+07], a controller that utilised both steering and braking was implemented in a co-simulation environment with Matlab/Simulink and Carsim 7. Some simulation results were shown that resemble understeer and oversteer situations. In both simulations the controller performed as expected. The main difficulties, however, occurred in how to choose the constraints (both for steering and braking) and how to deal with the non-linearities of the vehicle. It is possible to choose the constraints “loosely”, but then the linear model in the control allocation algorithm is not valid anymore. A possible method for dealing with the non-linearities is to make the model parameter dependent, like in Equation (4.2). It is also wise to make the constraints parameter dependent, since different driving conditions require different constraints. One can imagine that the constraint on maximum steering angle is different for steering at low and high speeds. Some sort of scenario estimator is therefore necessary.

The main non-linearities in the vehicle that were not modelled and that strongly influence the driving behavior are

- The tyre brake force is not directed parallel to the longitudinal axis of the vehicle, and the influence on yaw moment is therefore not obtained by simply multiplying the tyre brake force by half the track width.

- The tyre brake force is not directly a function of the brake moment, but a function of the longitudinal slip ratio $\kappa$, tyre side slip angle $\alpha$, vertical load $F_N$. The latter changes a lot in sharp corners since a lot of weight is placed on the wheels at the outer side of the corner and only a small part of the weight on the wheels at the inner side. In current ESC systems the longitudinal slip ratio $\kappa$ is used as actuator signal instead of the brake torque. The desired tyre slip ratio is fed to an ABS controller which eventually leads to a certain brake force at the tyres.

- The lateral tyre force is not a linear function of $\alpha$, but this relationship is strongly non-linear, see Figure 3.3.

- Compliance coefficients (suspension) and kingpin geometry (steering) also contribute to unmodelled phenomena. An example of this is the compliance coefficient that describes the toe angle versus longitudinal tyre force. During braking this coefficient tells the wheels to turn slightly inwards [Gil07]. This is not desired, but it is realistic.

- Actuators are not infinitely fast. In this chapter the actuator dynamics was not considered; it was not put in the model used for control allocation model nor was it used in the vehicle simulation model. The next chapter assumes non-negligible actuator dynamics and the optimisation algorithm is adjusted to deal with actuator dynamics.
Chapter 5

Control Allocation with Brake Dynamics

5.1 Introduction

In the previous chapter it was shown that it is possible to design and implement an electronic stability controller without the presence of actuator dynamics. Although actuator dynamics are fast compared to the car dynamics, neglecting them can lead to undesired behavior, see Figure 5.1. Here the static control approach of Chapter 4 is used with identical settings as used in Figure 4.5, but on a vehicle with first order brake dynamics with a time constant of 0.2 s.

![Figure 5.1: Achieved yaw velocity (left) and actuator inputs (cmd) and actuator output (act) signals (right) for an simulation with identical settings as Figure 4.5.](image)

The next section discusses the changes in the control allocation algorithm to incorporate actuator dynamics. After that simulation are performed for the 2-DOF model and the Carsim model of the BMW 330xi (Section 5.3).
5.2 Control Allocation

Since the relation between $y$ and $u_{cmd}$ is not static anymore, see Figure 4.2. The cost function given by Equation (4.7), which penalises the tracking error and control effort at the current time step $k$, needs to be extended to a cost function that penalises tracking error and control effort over a time horizon: $k, k + 1, k + 2, \ldots, k + H_p$, with $H_p$ the prediction horizon. The MPC controller optimises the cost function by calculating input trajectories over the entire horizon and then applies only the first input to the plant/vehicle. After that the optimisation is done again (receding horizon strategy) [Mac02].

In order to solve the problem using linear programming techniques and to penalise both the absolute value of controller output (for braking) and the changes in controller output (for steering), the cost function given by Equation (4.7) is used, but is adjusted such that it evaluates tracking error and control effort over the prediction horizon. The cost function then looks like

$$J(k) = \sum_{i=H_w}^{H_p} \|\dot{y}(k + i|k) - y_{des}(k + i|k)\|_1 + \lambda \sum_{j=0}^{H_u} \|W_p(\hat{u}_{cmd}(k + j|k) - u_p(k + j|k))\|_1,$$

subject to

$$\hat{u}_{cmd}(k + j|k) \leq u_{cmd}(k + j|k) \leq \overline{u}_{cmd}(k + j|k), \quad \text{for } j = 0, \ldots, H_u.$$

Here $\dot{y}(k + i|k)$ is the estimated yaw acceleration of the vehicle at future time steps $k + i$ for $i = H_w, \ldots, H_p$ for which the estimation is performed at current time step $k$. In a similar way $\hat{u}_{cmd}(k + j|k)$ represents the input signals that need to be supplied to the actuators at time steps $k + j$ for $j = 0, 1, \ldots, H_u$ to minimise the cost function. Again, these inputs are calculated at time step $k$. For the remainder of the report $H_u = H_p - 1$, and $H_w$ is set to be one for the ease of implementation. Setting $H_w = 1$ means that tracking error is not penalised at time step $k$, and setting $H_u = H_p - 1$ implies that controller effort is penalised from the current time step $k$ till $H_p - 1$.

To determine an estimate of the future yaw acceleration the MPC algorithm needs to use the mapping between actuator output and vehicle input, $G$, and the actuator model. There are several ways to estimate the future yaw acceleration, in this report it is done by using the step response model of the actuators, because (1) it is quite clear to understand, and (2) there is no need to calculate high powers of the system matrix, which could lead to numerical problems. Furthermore, the following assumptions are made

- The dynamics for both steering and braking is strictly proper, which is always true for physical systems.
- The actual actuator output at time step $k$ is available. In the simulation these outputs are simply measured to eliminate as many error sources as possible.
For a strictly proper SISO system with input $u$ and output $z$ the value after $H_p$ time steps can then be expressed as a function of variations in control input $\Delta \hat{u}(k+j\mid k) = \hat{u}(k+j\mid k) - \hat{u}(k+j-1\mid k)$\(^1\)

\[
\begin{align*}
\hat{z}(k+1\mid k) &= \hat{z}_f(k+1\mid k) + S(1)\Delta \hat{u}(k\mid k) \\
\hat{z}(k+H_p\mid k) &= \hat{z}_f(k+H_p\mid k) + S(H_p)\Delta \hat{u}(k\mid k) + S(H_p-1)\Delta \hat{u}(k+1\mid k) + \ldots \\
&\quad + S(2)\Delta \hat{u}(k + H_p - 2\mid k) + S(1)\Delta \hat{u}(k + H_p - 1\mid k).
\end{align*}
\] (5.2)

Here $\hat{z}_f(k+H_p\mid k)$ is the free response of the system when the future inputs would be kept constant at $u(k-1)$, and $S(i)$ are the step response parameters at time steps $k+i$.

The future yaw acceleration of the vehicle can be estimated in a similar way

\[
y = y_f + \Theta \Delta U,
\] (5.3)

where

\[
y = \begin{pmatrix}
\hat{y}(k+1\mid k) \\
\hat{y}(k+2\mid k) \\
\vdots \\
\hat{y}(k+H_p\mid k)
\end{pmatrix}, 
\quad y_f = \begin{pmatrix}
\hat{y}_f(k+1\mid k) \\
\hat{y}_f(k+2\mid k) \\
\vdots \\
\hat{y}_f(k+H_p\mid k)
\end{pmatrix}, 
\quad \Delta U = \begin{pmatrix}
\Delta \hat{u}(k\mid k) \\
\Delta \hat{u}(k+1\mid k) \\
\vdots \\
\Delta \hat{u}(k+H_p-1\mid k)
\end{pmatrix},
\]

and

\[
\Theta = \begin{pmatrix}
\text{GS}(1) & 0 & 0 & 0 & \cdots & 0 \\
\text{GS}(2) & \text{GS}(1) & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\text{GS}(H_p) & \text{GS}(H_p-1) & \ldots & \ldots & \ldots & \text{GS}(1)
\end{pmatrix}.
\] (5.4)

In the expression for $\Theta$ the step response matrices $S(i)$ are diagonal matrices with the step response values for the $n$ actuators at time step $k+i$ on the main diagonal. Since the yaw acceleration of the vehicle is modelled by the linear constant mapping $y = G\hat{u}_{act}$, all step response matrices need to be pre multiplied by $G$. To solve the optimisation problem in a similar fashion as in Chapter 4 Equation (5.3) needs to be rewritten such that it becomes a function of $\hat{u}(k+j)$

\[
\underbrace{y - y_f + Hu(k-1)}_{\tilde{y}_{des}} = \tilde{\Theta}U,
\] (5.5)

where

\^[1]Note that $\Delta \hat{u}(k\mid k) = \hat{u}(k\mid k) - u(k-1)$
\[
U = \begin{pmatrix}
\hat{u}(k|k) \\
\hat{u}(k+1|k) \\
\vdots \\
\hat{u}(k+H_p-1|k)
\end{pmatrix},
H = \begin{pmatrix}
\text{GS}(1) \\
\text{GS}(2) \\
\vdots \\
\text{GS}(H_p)
\end{pmatrix}
\]

and

\[
\tilde{\Theta} = \begin{pmatrix}
\text{GS}(1) & 0 & 0 & 0 & \ldots & 0 \\
G[S(2) - S(1)] & \text{GS}(1) & 0 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
G[S(H_p) - S(H_p - 1)] & G[S(H_p - 1) - S(H_p - 2)] & \ldots & \ldots & \ldots & \text{GS}(1)
\end{pmatrix}.
\]

(5.6)

The problem is now in a similar form as Equation (4.5) and can be solved in a similar way. The linear cost function then becomes

\[
\min_{x} J_d = \min_{x} \left[ 0 \ldots 0 \ 1 \ldots 1 \ \lambda \mathbf{w}_p^T \right] = \mathbf{m}^T \mathbf{x},
\]

subject to linear inequality constraints of the form \( \mathbf{A} \mathbf{x} \leq \mathbf{b} \), with \( \mathbf{A} \in \mathbb{R}^{4nH_p \times 3nH_p} \), and \( \mathbf{x}, \mathbf{b} \in \mathbb{R}^{3nH_p} \), with \( n \) the number of actuators.

### 5.3 Simulation results

In this section simulations are performed to test the new control allocation algorithm that includes actuator dynamics. The brake models used are a first order model with a time constant of \( \tau = 0.2 \) s, and a second order model of the form

\[
\frac{255}{s^2 + 30s + 255}.
\]

In Figure 5.2 both models are compared with a real electromechanical brake as used in [Lin07]. The dynamics of steering is neglected since the relation between the steering wheel and the actual wheel steer angle is kinematic and the only dynamics is in the tyre. The tire
has a “time” constant of 0.56 m for the lateral forces, which is equivalent to a time constant of $\tau \approx 0.014$ s when driving 144 km/h.

The first simulation performed is similar as the one shown in the introduction of this chapter, but then with the new dynamic control allocation algorithm. The prediction horizon $H_p$ is set to six time steps and the model of the brake dynamics used in the control allocation algorithm is identical to the brakes that are placed on the vehicle. The remaining settings are given in Table 4.1. Results are depicted in Figure 5.3 and it can be seen that the reference yaw velocity is tracked with the new control allocation algorithm. Looking at the applied brake signals one can see that the signals have larger brake input variations and that the signals oscillate more than with the static control allocation algorithm. It is not surprising that the brake input variations are larger than with the static control allocation algorithm since the control allocation algorithm takes the actuator dynamics into account. The actuator input, however, oscillates more than desired, which can also be seen in the input signal of the control allocation algorithm, see Figure 5.4. Two possible reasons for the oscillating behavior are

1. The incorrectness of the vehicle model used in the control allocation algorithm. Since the rotational dynamics is given by Equation (4.16) and (4.17) a large part of the vehicle dynamics is neglected.

2. The short prediction horizon. A prediction horizon of six time steps at 80 Hz means that the control allocation algorithm only looks $0.075$ s ahead. Figure 5.2 shows that the control allocation algorithm then only “knows” a small part of the brake dynamics. Unfortunately, it is impossible to increase the prediction horizon, because that leads to infeasible solutions (constraints are violated) when using Matlab’s linprog command to solve the optimisation problem.

Figure 5.5 shows the results for the same simulation, but with the second order brake model on the vehicle and in the control allocation algorithm. Clearly the desired yaw velocity is not tracked anymore. Again, a reason for this is that the control allocation algorithm does not
Figure 5.3: Simulation results for an understeer scenario with the Carsim vehicle model and first order actuator dynamics. In this scenario the lateral acceleration is approximately 0.85 g. Top left graph: desired and actual yaw velocity, top right graph: tracking error, bottom left graph: wheel steer angle, and bottom right graph: brake torques.

Figure 5.4: Input of the control allocation algorithm ($y_{des}$).
look far enough in the future. In addition, it may also have to do with how the free response
of Equation (5.3) is calculated. The calculate the free response for a state space system the
initial conditions for all states need to be supplied. If the brake model is of an order higher
than one, other states than the current brake torque (actuator output) are needed. At the
moment these extra states are simply set to zero.

![Simulation Results](image)

Figure 5.5: Simulation results for an understeer scenario with the Carsim vehicle model and
second order actuator dynamics. Top left: desired and actual yaw velocity, top right: tracking
error, bottom left: wheel steer angle, and bottom right: brake torques.

To exclude the effect of unmodelled dynamics in the control allocation algorithm two simu-
lations are performed with the vehicle simulated using the 2-DOF state space model. The
simulation parameters are adjusted, such that they are the same as in Table 4.1, meaning
that the vehicle is not allowed to steer through the entire manoeuvre but is forced to use its
brakes.

Figure 5.6 and Figure 5.7 show the results for a prediction horizon of six and three steps,
respectively. Two important conclusions can be drawn from Figure 5.6. First of all it confirms
that the unmodelled dynamics in the control allocation algorithm contribute to the oscillations
in Figure 5.3 and Figure 5.5, because with a correct vehicle model the oscillations are smaller
and the response is stable. Secondly, it shows that weights in the cost function are not chosen
correctly, since the wheel steer angle does not go to its upper bound when the tracking error
is zero.
By comparing Figure 5.6 and Figure 5.7 it can also be concluded that the length of the prediction horizon is very important, and definitely is one of the reasons for the oscillations and large tracking error. If the control allocation algorithm is unable to look far enough in the future (model error) the tracking is simply not good. Ideally, the control allocation algorithm should be able to look far enough ahead to know the brake dynamics. For the electromechanical brake of Figure 5.2 this should be around 0.2 s.

Figure 5.6: Simulation results for an understeer scenario with the 2-DOF vehicle model and second order actuator dynamics with a prediction horizon of six steps. Top left: desired and actual yaw velocity, top right: tracking error, bottom left: wheel steer angle, and bottom right: brake torques.
5.4 Conclusion

In this chapter the control allocation algorithm of Chapter 4 was extended to incorporate actuator dynamics. After that simulations were performed, whereby it was assumed that the dynamics of steering was much faster than the dynamics of the electromechanical brakes. The simulation results are summarised in Table 5.1.

Reasons for the oscillating tracking behaviour, and the unstable response for the second simulation are related to

- The incorrectness of the vehicle model used in the control allocation algorithm. The rotational dynamics is given by Equation (4.16) and (4.17) and therefore a very large part of the vehicle dynamics is neglected. Comparing the response of the second and third simulation supports this statement.

- The short prediction horizon. A prediction horizon of six time steps at 80 Hz means that the control allocation algorithm only looks 0.075 s ahead. Figure 5.2 shows that...
Table 5.1: Summary of simulation results with control allocation algorithm that incorporates brake dynamics

<table>
<thead>
<tr>
<th>Sim. No.</th>
<th>Brake model</th>
<th>Vehicle model</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>First order</td>
<td>Carsim model</td>
<td>Error in yaw velocity goes to zero.</td>
</tr>
<tr>
<td>2</td>
<td>Second order</td>
<td>Carsim model</td>
<td>Error in yaw velocity oscillates and the amplitude of the oscillation increases in time (unstable response).</td>
</tr>
<tr>
<td>3</td>
<td>Second order</td>
<td>2-DOF model</td>
<td>Error in yaw velocity goes to zero.</td>
</tr>
<tr>
<td>4</td>
<td>Second order\ ($H_p = 3$)</td>
<td>2-DOF model</td>
<td>Error in yaw velocity is oscillating near zero, but the response is stable.</td>
</tr>
</tbody>
</table>

the control allocation algorithm then only “knows” a small part of the brake dynamics. Comparing the response of the third and fourth simulation supports this statement. Unfortunately, it is impossible to increase the prediction horizon, because that leads to infeasible solutions (constraints are violated) when using Matlab’s `linprog` command to solve the optimisation problem.

- The way the free response in (5.3) is calculated. The calculate the free response for a state space system the initial conditions for all states need to be supplied. If the brake is of an order higher than one, other states than the current brake torque (actuator output) are needed. At the moment these extra states are simply set to zero, since information about these states is not readily available.

The next step could be to get the control allocation algorithm working properly for the 2-DOF vehicle model, meaning that the oscillations in the signals that are send to the brakes (command signals) should be small. This requires changing the optimisation algorithm which can be done in a lot of ways. Three of them are given here

1. Split the optimisation problem up in two parts, like in [LSY+05] and [LSY+07]. The problems that need to be solved are then smaller in size, which could be better than solving one large problem.

2. Use different software to solve the optimisation problem, for example by using the Multi-Parametric Toolbox. This is a free toolbox for Matlab for design, analysis and deployment of optimal controllers for constrained linear, nonlinear and hybrid systems. The user is also able to create his own cost function

3. Adjust the control allocation algorithm such that it is more flexible, i.e. no penalties in tracking error at every point $k + 1, \ldots, k + H_p$, and do not allow the actuator signal to vary at every point in the control horizon $k, \ldots, k + H_u$, but only at a few points. This leads to smaller constraint matrices, which should allow the prediction horizon to be set higher. Another option here is to introduce a high weight on the tracking error at $k = H_p$ (terminal cost term), which forces the tracking error to zero at the end of the prediction horizon.
Chapter 6

Summary and Recommendations

6.1 Summary

In this report part of an electronic stability controller for a full by-wire vehicle was developed that uses both steering and braking. Since designing a complete electronic stability controller was too much work for the available time, assumptions were made to put the focus on the yaw velocity controller and the distribution of control effort among redundant control actuators (control allocation). The most important assumptions were

- The desired yaw velocity was prescribed. Effectively this means that the driver’s commands are already interpreted. In [ZEP95] a more complete overview of an electronic stability controller can be found, which includes an interpreter for the driver’s commands and an estimator of the road friction coefficient.

- The road has a high coefficient of friction so there was no need for a vehicle side slip limiter.

In order to test the controller a realistic model of a vehicle was created. The test vehicle used in this report was a BMW 330xi from the year 2000. Chapter 3 described the creation of a realistic vehicle model in Carsim 7, which is a software program able to simulate the dynamic behavior of cars and light trucks. In addition, it is also able to put the vehicle through predefined manoeuvres and interface with other software programs, like Matlab/Simulink.

Chapter 4 described the use of model predictive control (MPC) theory for designing a (static) control allocation algorithm that utilises both steering and braking, under the assumption of infinitely fast actuator dynamics. In the vehicle model used for the control allocation the relation between actuator output (steering wheel angle and four brake torques) and vehicle input (yaw acceleration/moment) was assumed to be constant and linear. This relation was obtained using the bicycle model. The control allocation algorithm was tested in conjunction with a 2-DOF model identical to the vehicle model used in the MPC algorithm and on the - more realistic - Carsim model designed in Chapter 3. In both simulations the controller performed as expected. The main difficulties, however, occurred in how to set the constraints (both for steering and braking) and how to deal with the non-linearity of the vehicle. It
is possible to choose the constraints “loosely”, but then the linear model in the control allocation algorithm is not valid anymore. A possible method for dealing with the nonlinearities is to make the model parameter dependent, like in Equation (4.2). It is also wise to make the constraints parameter dependent, since different driving situations require different constraints. One can imagine that the constraint on steering angle is different for steering at low and high speeds. Some sort of scenario estimator is therefore necessary.

In Chapter 5 the control allocation algorithm was adjusted such that it was capable of dealing with actuator dynamics. Although actuator dynamics were relatively fast compared to vehicle dynamics, it was shown that it was not possible to use the static control allocation algorithm (of Chapter 4) when the vehicle model included actuator dynamics. After that simulations were performed where the brakes were modelled as first or second order systems, and where the dynamics of steering was neglected. Results of these simulations are summarised in Table 5.1. The main conclusion that can be drawn is that there are too many oscillations in the brake input signal, which sometimes lead to unstable behaviour. Two reasons for the oscillating behaviour are

1. The incorrectness of the vehicle model used in the control allocation algorithm. The rotational dynamics is given by Equation (4.16) and (4.17) and therefore a large part of the vehicle dynamics was neglected.

2. The short prediction horizon. If the control allocator is unable to “see” far enough in the future, it is unable to “see” all the dynamics and cannot calculate accurate future values of brake output. Unfortunately, it was impossible to increase the prediction horizon, because that leaded to infeasible solutions when using Matlab’s linprog command to solve the optimisation problem.

6.2 Recommendations

In the following some recommendations are made for further research.

1. The control allocation algorithm should work perfectly when the brake and vehicle model used in the allocation algorithm are the same as the actual brake and vehicle model. In that situation the oscillations in the signals that are send to the brakes (command signals) should be small or non existing. To achieve this the optimisation algorithm has to be changed, since it is not possible to choose a sufficiently large prediction horizon. Three possible changes which could lead to a better optimisation algorithm are:
   - Split the optimisation problem up in two parts, like in [LSY+05] and [LSY+07]. The problems that need to be solved are then smaller in size, which could be better than solving one large problem.
   - Use different software to solve the optimisation problem, for example by using the Multi-Parametric Toolbox. This is a free toolbox for Matlab for design, analysis and deployment of optimal controllers for constrained linear, nonlinear and hybrid systems.
Adjust the control allocation algorithm such that it is more flexible, i.e. no penalties in tracking error at every future point until the prediction horizon, and do not allow the actuator signal to vary at every point in the control horizon. This leads to smaller constraint matrices, which should allow the prediction horizon to be set higher. In addition, an inclusion of a terminal cost term that forces the tracking error to zero at the end of the prediction horizon might be helpful.

2. The model used in the control allocator is simple and maybe too simple. After improving the control allocation algorithm (previous point) the accuracy of the model should be investigated and a decision regarding the required level of detail of the vehicle model should be made. Possible improvement for the model are:

   - Include roll in the vehicle model.
   - Make use of tyre model that incorporate the combined effect of tyre side slip, longitudinal slip ratio, and (possible) vertical load. A nice example of this is shown in [And07].

3. Instead of prescribing the brake torques as vehicle input, the longitudinal tyre slip ratios should be prescribed. The slip ratios should then be given to the ABS controller, which makes sure that the tyres slip with the right amount, which eventually leads to a brake force.

4. A situation recognition element could be included, since different driving situations require different settings for constraints and weights in the optimisation algorithm. One can imagine that the maximum steering wheel angle at low speeds is larger than at high speeds, and that oversteer and understeer situations are also quite different. Although it might sometimes be possible to use the same weights for different situations. It can be very beneficial to use situation recognition and change the weights [And07].

5. A stability proof should be derived for the overall control scheme. At the moment a lot has been done by trial and error, since the test vehicle was only available as a complex non-linear carsim model. If, however, the vehicle is modelled as a linear system, the problem is similar to a classical cascaded control problem with two loops, and a stability proof could be derived more easy.

6. The control allocation algorithm can be extended with more actuators, like adjusting motor torque, individual wheel steer at all four wheels, or adjustment in the suspension.
Bibliography


## Appendix A

### List of Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Unit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_x$</td>
<td>m/s</td>
<td>Longitudinal velocity of vehicle</td>
</tr>
<tr>
<td>$v_y$</td>
<td>m/s</td>
<td>Lateral velocity of vehicle</td>
</tr>
<tr>
<td>$\dot{\psi}$</td>
<td>deg/s</td>
<td>Yaw velocity of vehicle</td>
</tr>
<tr>
<td>$V$</td>
<td>m/s</td>
<td>Velocity of vehicle</td>
</tr>
<tr>
<td>$\beta$</td>
<td>deg</td>
<td>Vehicle side slip angle</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>deg</td>
<td>Side slip angle of a tyre</td>
</tr>
<tr>
<td>$\alpha_f$</td>
<td>m/s</td>
<td>Tyre side slip angle at front wheel (in bicycle model)</td>
</tr>
<tr>
<td>$\alpha_r$</td>
<td>m/s</td>
<td>Tyre side slip angle at rear wheel (in bicycle model)</td>
</tr>
<tr>
<td>$\delta$</td>
<td>deg</td>
<td>Steering angle of the wheel</td>
</tr>
<tr>
<td>$R$</td>
<td>m</td>
<td>Cornering radius</td>
</tr>
<tr>
<td>$F_{y,f}$</td>
<td>N</td>
<td>Lateral tyre force at front wheel (in bicycle model)</td>
</tr>
<tr>
<td>$F_{y,r}$</td>
<td>N</td>
<td>Lateral tyre force at rear wheel (in bicycle model)</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>-</td>
<td>Longitudinal tyre slip ratio</td>
</tr>
<tr>
<td>$C$</td>
<td>N/deg</td>
<td>Cornering stiffness</td>
</tr>
<tr>
<td>$\mu$</td>
<td>-</td>
<td>Friction coefficient</td>
</tr>
<tr>
<td>$x_{ref}$</td>
<td>m</td>
<td>Reference trajectory</td>
</tr>
<tr>
<td>$T_b$</td>
<td>Nm</td>
<td>Brake torque</td>
</tr>
<tr>
<td>$T_{b,\dot{\psi}}$</td>
<td>Nm</td>
<td>Brake torque supplied by yaw velocity controller</td>
</tr>
<tr>
<td>$T_{b,\beta}$</td>
<td>Nm</td>
<td>Brake torque supplied by side slip controller</td>
</tr>
<tr>
<td>$\psi_{des}$</td>
<td>deg/s</td>
<td>Desired yaw velocity of vehicle</td>
</tr>
<tr>
<td>$\beta_{des}$</td>
<td>deg</td>
<td>Desired vehicle side slip angle</td>
</tr>
<tr>
<td>$\Delta \kappa$</td>
<td>-</td>
<td>Change in longitudinal tyre slip ratio</td>
</tr>
<tr>
<td>$F_{x,ij}$</td>
<td>N</td>
<td>Longitudinal tyre force</td>
</tr>
<tr>
<td>$F_{y,ij}$</td>
<td>N</td>
<td>Lateral tyre force</td>
</tr>
<tr>
<td>$F_z$</td>
<td>N</td>
<td>Vertical tyre force</td>
</tr>
<tr>
<td>$F_N$</td>
<td>N</td>
<td>Vertical tyre force</td>
</tr>
<tr>
<td>$M_z$</td>
<td>N</td>
<td>Self aligning torque on tyre</td>
</tr>
<tr>
<td>$\ddot{\psi}$</td>
<td>deg/s²</td>
<td>Yaw acceleration of vehicle</td>
</tr>
<tr>
<td>$\dot{\psi}_{des}$</td>
<td>deg/s²</td>
<td>Desired yaw acceleration of vehicle</td>
</tr>
<tr>
<td>$y_{des}$</td>
<td>deg/s²</td>
<td>Desired yaw acceleration of vehicle corrected</td>
</tr>
</tbody>
</table>
for acceleration due to the vehicle body

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u )</td>
<td>Vector containing actuator signals</td>
</tr>
<tr>
<td>( u_{cmd} )</td>
<td>Vector containing actuator input signals</td>
</tr>
<tr>
<td>( u_{act} )</td>
<td>Vector containing actuator output signals</td>
</tr>
<tr>
<td>( u_p )</td>
<td>Vector containing preferred actuator values</td>
</tr>
<tr>
<td>( u_s )</td>
<td>Vector of size ( u ) containing slack variables for tracking error</td>
</tr>
<tr>
<td>( u_s )</td>
<td>Vector of size ( u ) containing slack variables for control effort</td>
</tr>
<tr>
<td>( \dot{u} )</td>
<td>Vector containing the rate of change of the actuator signals</td>
</tr>
<tr>
<td>( \omega )</td>
<td>Vector containing the angular velocities (Roll, Pitch, Yaw)</td>
</tr>
<tr>
<td>( \dot{\omega} )</td>
<td>Vector containing the angular accelerations (Roll, Pitch, Yaw)</td>
</tr>
<tr>
<td>( \theta )</td>
<td>Vector containing parameters needed in the angular equation of motion</td>
</tr>
<tr>
<td>( n )</td>
<td>Number of control actuators</td>
</tr>
<tr>
<td>( W_p )</td>
<td>Matrix containing weight factors for the different actuators</td>
</tr>
<tr>
<td>( w_p )</td>
<td>Vector containing diagonal factors of ( W_p )</td>
</tr>
<tr>
<td>( \lambda )</td>
<td>Weighting factor between tracking error and control effort</td>
</tr>
<tr>
<td>( T_s )</td>
<td>Sample time</td>
</tr>
<tr>
<td>( H_p )</td>
<td>Length of prediction horizon</td>
</tr>
<tr>
<td>( H_w )</td>
<td>First point in prediction horizon at which tracking error in penalised</td>
</tr>
<tr>
<td>( H_w )</td>
<td>Control horizon</td>
</tr>
<tr>
<td>( \Delta u )</td>
<td>Variation in control input</td>
</tr>
<tr>
<td>( \tau )</td>
<td>Time constant</td>
</tr>
</tbody>
</table>
Appendix B

Simulation setup

In the first section a road map to set up a co-simulation environment between Matlab/Simulink and Carsim 7 is given. It is thereby assumed that the reader is familiar with the basics of Carsim and has access to all the Carsim manuals. After that the Simulink files (Section B.2) and M-files (Section B.3) used in this report are given. Finally, the equations of motion of the 2-DOF vehicle model are also given (Section B.4).

B.1 Setting up the simulation environment

Get Carsim 7 up and running Carsim 7 requires a license for the major components of the program. For just designing a vehicle model it is sufficient to only have the the license for the “CarSim Simulation Graphic User Interface” (SGUI). If, however, simulations have to be performed using the build-in solver (VS solver), a separate license is needed. Most of the time these licenses are so called “network licenses” and are run from a central license server.

Model the vehicle and the environment The vehicle - as described in Chapter 3 - can now be modelled according to the Carsim manual. After that the environment can be set and a predefined manoeuvre can be chosen. An other options is to import all the relevant data from the .par files if they are available. In that way models and environmental settings are loaded directly into Carsim. For this report these .par files are available and it is recommended to load in the settings of “CarSim Events and procedures.par” for all the manoeuvres and “CarSim BMW330xi.par” for the vehicle models. The latter file contains six vehicle models, which are all stored in the dataset: “double lane change”. The six models are

- The BMW as modelled in [HD07]. Filename: DLC: BMW Original model.
- The reference C-class hatchback car used for model verification and to extract unknown parameters. Filename: DLC: Reference C-class HB.

1Several licenses are normally put in one file, so it can be that one file contains all the licenses for a whole company.
The reference rear wheel drive D-class sedan, which was also used for model verification and to extract unknown parameters. Filename: DLC: Reference RWD D-class Sedan.

- The BMW as modelled in Chapter 3, but than with simplified suspension and steering. Filename: DLC: BMW Adjusted Sedan Simplified

- *The BMW as modelled in Chapter 3. Filename: DLC: BMW Adjusted Sedan*

- The BMW as modelled in Chapter 3, but than with the suspension modelled using the old modelling screen and settings from Carsim 5. Filename: DLC: BMW Adjusted Sedan Simple Suspension.

**Setup the link between Carsim 7 and Simulink** Instead of simulating with the build in solver (VS Solver) it is also possible to use other software, like Simulink. To setup the co-simulation environment the user has the change the option “Run Control: Built-In Solvers” to “Run Control with Simulink”. After that the user has to pick an existing dataset or create a new one. If a new one has to be created the user has to specify the import and output settings of the vehicle and the location of an existing Simulink model, to which Carsim sends the equations of motion that have to be solved. If these datasets are imported via .par files, the user has to make sure that the location of the Simulink files is set correctly. For this report four different datasets are used (see Section C.2).

**Running a simulation** The simulation can now be started by either pressing the “Run from Here” button or by pressing the “Send to Simulink” button. In the first scenario the accompanying Simulink model is opened, the simulation is then started automatically in Simulink, and finally the Simulink model is closed and the post processing can be done in Carsim. In the second scenario Carsim only opens the accompanying Simulink model, and the user is able to control it from there. To user is allowed to make changes in the Simulink file, or run several simulations. Post processing can be done in Matlab or Simulink, but also in Carsim.

The import channels that are used in this report are given in Table B.1. The lateral tyre forces are used to create an oversteer situation. If “Mode” is set to “Replace” the internal variable is replaced by the externally supplied variable. If “Mode” is set to “Add” the supplied value is added to the value of the internal value.

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter name</th>
<th>Mode</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Steering wheel angle</td>
<td>IMP_STEER_SW</td>
<td>Replace</td>
<td>0.0</td>
</tr>
<tr>
<td>Brake torque left front wheel</td>
<td>IMP_MYBK_L1</td>
<td>Replace</td>
<td>0.0</td>
</tr>
<tr>
<td>Brake torque left rear wheel</td>
<td>IMP_MYBK_L2</td>
<td>Replace</td>
<td>0.0</td>
</tr>
<tr>
<td>Brake torque right front wheel</td>
<td>IMP_MYBK_R1</td>
<td>Replace</td>
<td>0.0</td>
</tr>
<tr>
<td>Brake torque right rear</td>
<td>IMP_MYBK_R1</td>
<td>Replace</td>
<td>0.0</td>
</tr>
<tr>
<td>Transmission output shaft torque</td>
<td>IMP_M_OUT_TR</td>
<td>Add</td>
<td>0.0</td>
</tr>
<tr>
<td>Lateral force on left rear tyre</td>
<td>IMP_FY_L2</td>
<td>Add</td>
<td>0.0</td>
</tr>
<tr>
<td>Lateral force on right rear tyre</td>
<td>IMP_FY_R2</td>
<td>Add</td>
<td>0.0</td>
</tr>
</tbody>
</table>

The import channels that are used in this report are given in Table B.1. The lateral tyre forces are used to create an oversteer situation. If “Mode” is set to “Replace” the internal variable is replaced by the externally supplied variable. If “Mode” is set to “Add” the supplied value is added to the value of the internal value.
In a similar way the output variables are given in Table B.2. The majority of these variables are not used by the electronic stability controller, but are used for analysis only.

B.2 Simulink models

In this section and the next the Simulink files and M-files used to perform the simulation are given. In total, six different Simulink models are used, which also make use of different M-files to extract parameters and run the optimisation algorithm. In addition, each set of files is used in conjunction with a dataset in the “Run Control with Simulink” field of Carsim. In the following an overview is given of how these files are all linked together.

BMWsimulink_io_static_understeer_skateboard.mdl, see Figure B.1
Carsim involvement: not needed.
The first simulation setup is used to test the static control allocation algorithm (no actuator dynamics) on a simple 2-DOF car model, which is identical to the model used in the control allocation algorithm. Carsim involvement is therefore not needed, since a state space system block is used to implement the 2-DOF model. The M-files needed to run the simulation are

• init.m
• parameters_static_skateboard.m
• optimisation_static.m

BMWsimulink_io_static_understeer_final.mdl
Carsim involvement: Simulink model io - Static - Understeer.
The second simulation setup is similar to the first, but the 2-DOF model is replaced by the Carsim model. The M-files needed to run the simulation are

• init.m
• parameters_static.m
• optimisation_static.m

BMWsimulink_io_static_oversteer_final.mdl
Carsim involvement: Simulink model io - Static - Oversteer.
The third simulation is similar to the second, but here an oversteer situation is simulated. The needed M-files are

• init.m
• parameters_static.m
• optimisation_static_oversteer.m
BMWsimulink_io_static_vs_dynamic_understeer_final.mdl
Carsim involvement: Simulink model io - Static Dynamic Comp - Understeer.
The fourth setup is used to simulate the static control allocation algorithm on a vehicle with brake dynamics. The needed M-files are

- init.m
- parameters_dynamic.m
- optimisation_static.m

BMWsimulink_io_dynamic_understeer_skateboard.mdl
Carsim involvement: not needed.
The fifth setup is used to simulate the dynamic control allocation algorithm on the 2-DOF model with brake dynamics. Carsim is again not needed and the needed M-files are

- init.m
- parameters_dynamic_skateboard.m
- optimisation_dynamic.m

BMWsimulink_io_dynamic_understeer.mdl, see Figure B.2
Carsim involvement: Simulink model io - Dynamic - Understeer
The last setup is used to simulate the dynamic control allocation algorithm on the Carsim model with brake dynamics. The needed M-files are

- init.m
- parameters_dynamic.m
- optimisation_dynamic.m
### Table B.2: Output channels

<table>
<thead>
<tr>
<th>Description</th>
<th>Parameter name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Longitudinal speed of left front tyre</td>
<td>Vx_L1</td>
</tr>
<tr>
<td>Longitudinal speed of left rear tyre</td>
<td>Vx_L2</td>
</tr>
<tr>
<td>Longitudinal speed of right front tyre</td>
<td>Vx_R1</td>
</tr>
<tr>
<td>Longitudinal speed of right rear tyre</td>
<td>Vx_R2</td>
</tr>
<tr>
<td>Lateral speed of left front tyre (at tyre contact point)</td>
<td>VyTC_L1</td>
</tr>
<tr>
<td>Lateral speed of left rear tyre (at tyre contact point)</td>
<td>VyTC_L2</td>
</tr>
<tr>
<td>Lateral speed of right front tyre (at tyre contact point)</td>
<td>VyTC_R1</td>
</tr>
<tr>
<td>Lateral speed of right rear tyre (at tyre contact point)</td>
<td>VyTC_R2</td>
</tr>
<tr>
<td>Longitudinal tyre slip at left front wheel</td>
<td>Kappa_L1</td>
</tr>
<tr>
<td>Longitudinal tyre slip at left rear wheel</td>
<td>Kappa_L2</td>
</tr>
<tr>
<td>Longitudinal tyre slip at right front wheel</td>
<td>Kappa_R1</td>
</tr>
<tr>
<td>Longitudinal tyre slip at right rear wheel</td>
<td>Kappa_R2</td>
</tr>
<tr>
<td>Tyre side slip angle at left front tyre</td>
<td>Alpha_L1</td>
</tr>
<tr>
<td>Tyre side slip angle at left rear tyre</td>
<td>Alpha_L2</td>
</tr>
<tr>
<td>Tyre side slip angle at right front tyre</td>
<td>Alpha_R1</td>
</tr>
<tr>
<td>Tyre side slip angle at right rear tyre</td>
<td>Alpha_R2</td>
</tr>
<tr>
<td>Wheel steer angle at left front wheel</td>
<td>Steer_L1</td>
</tr>
<tr>
<td>Wheel steer angle at right front wheel</td>
<td>Steer_R1</td>
</tr>
<tr>
<td>Vehicle side slip angle</td>
<td>Beta</td>
</tr>
<tr>
<td>Longitudinal vehicle speed</td>
<td>Vx</td>
</tr>
<tr>
<td>Lateral vehicle speed</td>
<td>Vy</td>
</tr>
<tr>
<td>Vertical vehicle speed</td>
<td>Vz</td>
</tr>
<tr>
<td>Longitudinal acceleration of vehicle</td>
<td>Ax</td>
</tr>
<tr>
<td>Lateral acceleration of vehicle</td>
<td>Ay</td>
</tr>
<tr>
<td>Vertical acceleration of vehicle</td>
<td>Az</td>
</tr>
<tr>
<td>Roll velocity of the sprung mass of the vehicle</td>
<td>AVx</td>
</tr>
<tr>
<td>Pitch velocity of the sprung mass of the vehicle</td>
<td>AVy</td>
</tr>
<tr>
<td>Yaw velocity of the sprung mass of the vehicle</td>
<td>AVz</td>
</tr>
<tr>
<td>Roll acceleration of the sprung mass of the vehicle</td>
<td>AAx</td>
</tr>
<tr>
<td>Pitch acceleration of the sprung mass of the vehicle</td>
<td>AAY</td>
</tr>
<tr>
<td>Yaw acceleration of the sprung mass of the vehicle</td>
<td>AAz</td>
</tr>
<tr>
<td>Longitudinal tyre force at left front tyre</td>
<td>Fx_L1</td>
</tr>
<tr>
<td>Longitudinal tyre force at left rear tyre</td>
<td>Fx_L2</td>
</tr>
<tr>
<td>Longitudinal tyre force at right front tyre</td>
<td>Fx_R1</td>
</tr>
<tr>
<td>Longitudinal tyre force at right rear tyre</td>
<td>Fx_R2</td>
</tr>
<tr>
<td>Lateral tyre force at left front tyre</td>
<td>Fy_L1</td>
</tr>
<tr>
<td>Lateral tyre force at left rear tyre</td>
<td>Fy_L2</td>
</tr>
<tr>
<td>Lateral tyre force at right front tyre</td>
<td>Fy_R1</td>
</tr>
<tr>
<td>Lateral tyre force at right rear tyre</td>
<td>Fy_R2</td>
</tr>
<tr>
<td>Vertical tyre force at left front tyre</td>
<td>Fz_L1</td>
</tr>
<tr>
<td>Vertical tyre force at left rear tyre</td>
<td>Fz_L2</td>
</tr>
<tr>
<td>Vertical tyre force at right front tyre</td>
<td>Fz_R1</td>
</tr>
<tr>
<td>Vertical tyre force at right rear tyre</td>
<td>Fz_R2</td>
</tr>
</tbody>
</table>
Figure B.1: Simulink model used to test the static control allocation algorithm on the 2-DOF vehicle model (Chapter 4). The subsystems “Dynamic Inversion” and “Ref generator” are depicted in Figure B.3 and Figure B.4, respectively.
Figure B.2: Simulink model used to test the dynamic control allocation algorithm on the CarSim vehicle model (Chapter 5). The subsystems “Dynamic Inversion”, “Ref generator”, and “Act dynamics” are depicted in Figure B.3, Figure B.4, and Figure B.5, respectively.
Figure B.3: Simulink model of the dynamic inversion block in Figure B.1 and B.2.

Figure B.4: Simulink model of the reference generator in Figure B.1 and B.2.
Figure B.5: Simulink model of the Act. dynamics block in Figure B.2.
B.3 M-files

Init.m
clear all

Parameters_static_skateboard.m

%% This file contains parameters needed by the simulation files to run,  
%% in the case that actuator dynamics is neglected and a 2nd order  
%% state space model is used to simulate the vehicle (skateboard model).

%% Simulation parameters
Fs = 80;    % Sampling frequency [Hz]
Ts = 1/Fs;

%% Car properties
J = 2730;    % Moment of Inertia around z-acis [kgm2]
m = 1725;    % Vehicle mass [kg]

%% Car dimensions
a = 1.365;    % Distance front axle to CoG [m]
b = 1.360;    % Distance rear axle to CoG [m]
t_f = 1.471;  % Track width front [m]
t_r = 1.478;  % Track width rear [m]
rdyn = 0.3184; % Dynamic Tyre radius [m]

%% Tyre parameters
Cf = 1600*180/pi;  % Stiffness front tyres [N/rad]
Cr = 1600*180/pi;  % Stiffness rear tyres [N/rad]

%% Vehicle model
u = 40;        % m/s vehicle speed

% x1 = v (lateral velocity)
% x2 = r (yaw velocity)

Aveh = [-2/u*(Cf + Cr)/m -(u + 2/u*(a*Cf - b*Cr))/m;  
    -2/u*(a*Cf - b*Cr)/J -2/u*(Cf*a^2 + Cr*b^2)/J];
Bveh = [2*Cf/m 0 0 0 0;  
    2*Cf*a/J t_f/2/rdyn/J t_r/2/rdyn/J -t_f/2/rdyn/J -t_r/2/rdyn/J];
Cveh = eye(2);
Dveh = [0 0 0 0 0;0 0 0 0 0];
sys = ss(Aveh,Bveh,Cveh,Dveh);

Parameters_static.m

%% This file contains parameters needed by the simulation files to run,  
%% in the case that actuator dynamics is neglected
%% Simulation parameters
Fs = 80;  \% Sampling frequency [Hz]
Ts = 1/Fs;

%% Car properties
J = 2730;  \% Moment of Inertia around z-axis [kgm^2]

%% Car dimensions
a = 1.365;  \% Distance front axle to CoG [m]
b = 1.360;  \% Distance rear axle to CoG [m]
t_f = 1.471;  \% Track width front [m]
t_r = 1.478;  \% Track width rear [m]
rdyn = 0.3184;  \% Dynamic Tyre radius [m]

%% Tyre parameters
Cf = 1600*180/pi;  \% Stiffness front tyres [N/deg]
Cr = 1600*180/pi;  \% Stiffness rear tyres [N/deg]

Parameters_dynamic_skateboard.m

%% This file contains parameters needed by the simulation files to run,
%% in the case that actuator dynamics is neglected and a 2nd order
%% state space model is used to simulate the vehicle (skateboard model).

%% Control horizon
Hp = 6;

%% Simulation parameters
Fs = 80;  \% Sampling frequency [Hz]
Ts = 1/Fs;

%% Brake dynamics
% Third order model
num = [10950];
den = [1 36 994 10950];

% Fast second order model
% num = [1000];
% den = [1 70 1000];

% Realistic second order model
% num = [255];
% den = [1 30 255];

% First order model - slow: tau = 0.2 s
% num = [5];
% den = [1 5];

% First order model - fast: tau = 0.01 s
% num = [100];
% den = [1 100];
%% Car properties
J = 2730; % Moment of Inertia around z-axis [kgm²]
m = 1725; % Vehicle mass [kg]

%% Car dimensions
a = 1.365; % Distance front axle to CoG [m]
b = 1.360; % Distance rear axle to CoG [m]
t_f = 1.471; % Track width front [m]
t_r = 1.478; % Track width rear [m]
rdyn = 0.3184; % Dynamic Tyre radius [m]

%% Tyre parameters
Cf = 1600*180/pi; % Stiffness front tyres [N/rad]
Cr = 1600*180/pi; % Stiffness rear tyres [N/rad]

%% Vehicle model
u = 40; % m/s vehicle speed

% x1 = v (lateral velocity)
% x2 = r (yaw velocity)
Aveh = [-2/u*(Cf + Cr)/m -(u + 2/u*(a*Cf - b*Cr)/m)/m;
      -2/u*(a*Cf - b*Cr)/J -2/u*(Cf*a^2 + Cr*b^2)/J];
Bveh = [2*Cf/m 0 0 0 0;
       2*Cf*a/J t_f/2/rdyn/J t_r/2/rdyn/J -t_f/2/rdyn/J -t_r/2/rdyn/J];
Cveh = eye(2);
Dveh = [0 0 0 0 0;0 0 0 0 0];
sys = ss(Aveh,Bveh,Cveh,Dveh);

Parameters_dynamic.m

%% This file contains parameters needed by the simulation files to run,
%% in the case that actuator dynamics is included

%% Control horizon
Hp = 6;

%% Simulation parameters
Fs = 80; % Sampling frequency [Hz]
Ts = 1/Fs;

%% Brake dynamics
% Third order model
% num = [10950];
% den = [1 36 994 10950];

% Fast second order model
% num = [1000];
% den = [1 70 1000];

% Realistic second order model
\% num = [255];
\% den = [1 30 255];

\% First order model - slow: \( \tau = 0.2 \, \text{s} \)
num = [5];
den = [1 5];

\% First order model - fast: \( \tau = 0.01 \, \text{s} \)
\% num = [100];
\% den = [1 100];

%% Car properties
J = 2730; \quad \% \text{Moment of Inertia around z-axis [kgm}^2]\%

\% Car dimensions
a = 1.365; \quad \% \text{Distance front axle to CoG [m]}
b = 1.360; \quad \% \text{Distance rear axle to CoG [m]}
t_f = 1.471; \quad \% \text{Track width front [m]}
t_r = 1.478; \quad \% \text{Track width rear [m]}
rdyn = 0.3184; \quad \% \text{Dynamic Tyre radius [m]}

%% Tyre parameters
C_f = 1600*180/pi; \quad \% \text{Stiffness front tyres [N/deg]}
C_r = 1600*180/pi; \quad \% \text{Stiffness rear tyres [N/deg]}

Optimisation_static.m & Optimisation_static_oversteer.m

\% Optimisation - No actuator dynamics FINAL (Phase 1)
\% Weight are set for an understeer situation

\% This M-file is used to perform the control allocation in phase 1
\% Here a static relationship between actuator outputs and vehicle
\% inputs is assumed as well as inf. fast actuator dynamics.

\% inputs:
\% \text{y\_des: yaw acc to be tracked in rad/s}^2
\% \text{delta\_p: actuator output at previous timestep}

\% output:
\% \text{\textup{delta}: new actuator outputs:}
\% \text{[Delta\_nom\_steerin\_wheel (deg), Tlf (Nm), Tlr (nm), Trf (nm), Trr (nm)]}

\% Additional files needed:
\% * parameters_static File contains car parameters

function [out] = optimisation(in)

y\_des = in(1);
delta\_p = in(2:6);

\% Load car properties
parameters_static
% Bounds on actuator output
steer_max = 2*16.8;  % deg at steering wheel
Tf_max = 1000;
Tr_max = 900;
rate_steer = 900;  % deg/s at steering wheel
rate_brake = 200000;

% Weights

% Relative weight of control action vs tracking error
lambda = 0.3;
% weights on [d_sw Tf_l Trf Trf Trr]
Wp_col = [1 2e0 1e0 2e0 1e0];
Wp = diag([Wp_col]);
scale = [16.8 1000 1000 1000 1000];  % Scale factor

ub = [steer_max Tf_max Tr_max Tf_max Tr_max 1000*ones(1,10)]';
ubr = [delta_p(1)+rate_steer*Ts;delta_p(2:5)+rate_brake*Ts;1000*ones(10,1)];
ub = min(ub,ubr);
ub(1:5,1) = 1./scale'.*ub(1:5,1);

lb = [-steer_max 0 0 0 0 zeros(1,10)]';
lbr = [delta_p(1)-rate_steer*Ts;delta_p(2:5)-rate_brake*Ts;zeros(10,1)];
lb = max(lb,lbr);
lb(1:5,1) = 1./scale'.*lb(1:5,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%% Linear programming problem %%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Relation between vehicle inputs and plant outputs: y = G*delta
% delta = [delta_steering wheel (rad) Brake torques (Nm)]
G = 1./J.*scale.*[2*(Cf*pi/180)*a/16.8 t_f/2/rdyn t_r/2/rdyn -t_f/2/rdyn -t_r/2/rdyn];

%% Handy variables:
nil = zeros(5,5);
one = eye(5);
min_one = -1*one;
Gbar = [G;G;G;G;G];
Y_des = [y_des;y_des;y_des;y_des;y_des];

%% Optimisation implementation

%% Optimisation cost function
f = [zeros(1,5) ones(1,5) lambda*Wp_col]';

%% Inequality constraints
A = [Gbar min_one nil; -1*Gbar min_one nil
     one nil min_one;];
min_one nil min_one];

B = [Y_des;
    -Y_des;
    delta_p(1)/scale(1);
    zeros(4,1);
    -delta_p(1)/scale(1);
    -zeros(4,1)];

%% Solving the optimisation problem

% Setting default options for linprog-algorithm:
options = optimset('linprog');

% Set (additional) options:
options = optimset(options,'LargeScale','off','Display','Off');
init = [delta_p; zeros(10,1)];
[delta, fval] = linprog(f,A,B,[],[],lb,ub,init,options);

%% Output
out = [fval; scale'.*delta(1:5,1)];

 Optimisation_dynamic

%% Optimisation with actuator dynamics FINAL (Phase 2)
%% Weight are set for an understeer situation

%% This Mfile is used to perform the control allocation in phase r
%% Here a dynamic relation between actuator input and output
%% is assumed

%% inputs:
% y_des: yaw acc to be tracked in rad/s^2
% u_old: actuator input at previous timestep
% u_act: actuator output at previous timestep

%% output:
% delta: new actuator outputs:
% [Delta_nom_steerin_wheel (deg), Tlf (Nm), Tlr (nm), Trf (nm), Trr (nm)]

%% Additional files needed:
% * none

function [out] = MPCoptimisation(in)
% in = [10;0;0;0;0;0;0;0;0;0;0;3];

%% Bounds on actuator input
steer_max = 0.5*16.8;
Tf_max = 1000;
Tr_max = 900;

%% Weights
% Relative weight of control action vs tracking error
lambda = 0.3;

% weights on [d_sw Tlf Tlr Trf Trr]
Wp_row = [1 2e0 1e0 2e0 1e0];

%% inputs
scale = [16.8 1000 1000 1000 1000];
setpoint = in(1);
u_old = in(2:6)./scale';
u_act = in(7:11)./scale';

%% Simulation parameters
Fs = 80; % Sampling frequency [Hz]
Ts = 1/Fs;

%% Optimisation properties
Hp = in(12);
simtime = 0:Ts:Hp*Ts;

%% Model of brake dynamics

% Third order model
% num = [10950];
% den = [1 36 994 10950];
% Asys = [-36 -994 -10950; 1 0 0; 1 0];
% Bsys = [10950; 0; 0];
% Csys = [0 0 1];
% Dsys = 0;
% orde = 3;

% Realistic second order model
% num = [255];
% den = [1 30 255];
% Asys = [-30 -255; 1 0];
% Bsys = [255 0]';
% Csys = [0 1];
% Dsys = 0;
% orde = 2;

% First order model - fast: tau = 0.01 s
% num = [100];
% den = [1 100];
% Asys = -100;
% Bsys = 100;
% Csys = 1;
% Dsys = 0;
% orde = 1;

% First order model - slow: tau = 0.2 s
num = [5];
den = [1 5];
Asys = -5;
Bsys = 5;
Csys = 1;
Dsys = 0;
orde = 1;

%% Create sys object
sys = ss(Asys,Bsys,Csys,Dsys);

%% Set initial conditions to calculate free response
X0Tblf = [zeros(1,orde-1) u_act(2)]';
X0Tblr = [zeros(1,orde-1) u_act(3)]';
X0Tbrf = [zeros(1,orde-1) u_act(4)]';
X0Tbrr = [zeros(1,orde-1) u_act(5)]';

%% Car properties
J = 2730;  % Moment of Inertia around z-acis [kgm2]
a = 1.365;  % Distance front axle to CoG [m]
b = 1.360;  % Distance rear axle to CoG [m]
t_f = 1.471;  % Track width front [m]
t_r = 1.478;  % Track width rear [m]
rdyn = 0.3184;  % Dynamic Tyre radius [m]

%% Tyre parameters
Cf = 1600*180/pi;  % Stiffness front tyres [N/deg]
Cr = 1600*180/pi;  % Stiffness rear tyres [N/deg]

%% Set setpoint
Ref = setpoint*ones(Hp,1);

ub = [steer_max Tf_max Tr_max Tf_max Tr_max]';
ub = 1./scale'.*ub;

lb = [-steer_max 0 0 0 0]';
lb = 1./scale'.*lb;

%% Dummies in order to construct the constraint matrices to solve
%% the optimisation problem as a linear programming problem
U0 = [ ]; UB = [ ]; LB = [ ]; RR = [ ]; SS = [ ];
WPR = [ ]; Aperf1 = zeros(5*Hp,3*5*Hp); min_one = -diag([1 0 0 0 0]);
Aperf2 = zeros(5*Hp,3*5*Hp); ERR = [ ];

DP = zeros(5*Hp,1);  % corresponding to delta_p
DP(1,1) = u_old(1);  % Other parts are put in Aeq

for i = 1:Hp
    WPR = [WPR Wp_row];
end
% WPR(1:1:15) = [0 4 2 4 2 0 4 2 4 2 0 2 1 2 1];

for i = 1:Hp
    U0 = [U0; u_old];
    UB = [UB; ub];
LB = [LB; lb];
end
U0 = [U0; zeros(5*Hp,1); zeros(5*Hp,1)];
UB = [UB; 10000*ones(5*Hp,1); 10000*ones(5*Hp,1)];
LB = [LB; zeros(5*Hp,1); zeros(5*Hp,1)];

Aperf1(1:5*Hp,1:5*Hp) = eye(5*Hp);
Aperf1(1:5*Hp,end-5*Hp+1:end) = -eye(5*Hp);
Aperf2(1:5*Hp,1:5*Hp) = -eye(5*Hp);
Aperf2(1:5*Hp,end-5*Hp+1:end) = -eye(5*Hp);

for i=1:Hp-1
    Aperf1(5*(i+1)-4:5*(i+1),5*i-4:5*i) = min_one;
    Aperf2(5*(i+1)-4:5*(i+1),5*i-4:5*i) = -min_one;
end

%% Linear relation between actuator output and plant input
G = 1/J.*scale.*[2*(Cf*pi/180)*a/16.8 t_f/2/rdyn t_r/2/rdyn -t_f/2/rdyn -t_r/2/rdyn];

%% Calculate free response
Ufree_Tblf = ones(1,length(simtime))*u_old(2);
Ufree_Tblr = ones(1,length(simtime))*u_old(3);
Ufree_Tbrf = ones(1,length(simtime))*u_old(4);
Ufree_Tbrr = ones(1,length(simtime))*u_old(5);

Tblf_free = lsim(sys,Ufree_Tblf,simtime,X0Tblf);
Tblr_free = lsim(sys,Ufree_Tblr,simtime,X0Tblr);
Tbrf_free = lsim(sys,Ufree_Tbrf,simtime,X0Tbrf);
Tbrr_free = lsim(sys,Ufree_Tbrr,simtime,X0Tbrr);

Yf = ones(5*length(simtime),1);
Yf(1:5:length(Yf),1) = u_old(1);
Yf(2:5:length(Yf),1) = Tblf_free';
Yf(3:5:length(Yf),1) = Tblr_free';
Yf(4:5:length(Yf),1) = Tbrf_free';
Yf(5:5:length(Yf),1) = Tbrr_free';

GG = ones(length(simtime),5);
Yf_tot = ones(length(simtime),1);

for i=1:length(simtime)
    GG(i,:) = G;
end

for i=1:length(simtime)
    Yf_tot(i,1) = GG(i,1:end) * Yf(5*i-4:5*i);
end

%% Step response
Dswstep = ones(length(simtime),1)*1;
Tbstep = step(sys,simtime);

% NB: S(0) = Tbstep(1)
Theta = zeros(5*Hp,5*Hp);
for i=1:Hp
    Theta(5*i-4:5*i,5*i-4:5*i) = diag([Dswstep(i+1) Tbstep(i+1)*ones(1,4)]);
end

Theta_bar = zeros(Hp,5);
for i=1:Hp
    Theta_bar(i,:) = G*Theta(5*i-4:5*i,5*i-4:5*i);
end

B = zeros(Hp,5*(Hp));
for j = 0:Hp-1
    for i = 1:Hp-j
        if j == 0
            B(i+j,5*i-4:5*i) = Theta_bar(j+1,:);
        else
            B(i+j,5*i-4:5*i) = Theta_bar(j+1,:) - Theta_bar(j,:);
        end
    end
end

Slack1 = zeros(Hp,5*Hp);
Slack2 = zeros(Hp,5*Hp);
Slack3 = zeros(Hp,5*Hp);
Slack4 = zeros(Hp,5*Hp);
Slack5 = zeros(Hp,5*Hp);
for i = 1:Hp
    Slack1(i,5*i-4) = -1;
    Slack2(i,5*i-3) = -1;
    Slack3(i,5*i-2) = -1;
    Slack4(i,5*i-1) = -1;
    Slack5(i,5*i) = -1;
end

% [delta(k-1) delta(k) delta_p delta_sw]
% f = [zeros(1,5*(Hp)) ones(1,5*(Hp)) lambda*WPR];

Aineq = [B Slack1 zeros(Hp,5*Hp); B Slack2 zeros(Hp,5*Hp); B Slack3 zeros(Hp,5*Hp); B Slack4 zeros(Hp,5*Hp); B Slack5 zeros(Hp,5*Hp); -B Slack1 zeros(Hp,5*Hp); -B Slack2 zeros(Hp,5*Hp); -B Slack3 zeros(Hp,5*Hp); -B Slack4 zeros(Hp,5*Hp); -B Slack5 zeros(Hp,5*Hp); Aperf1; Aperf2];
Bineq = [Ref - Yf_tot(2:end) + Theta_bar*u_old;
    Ref - Yf_tot(2:end) + Theta_bar*u_old;
    Ref - Yf_tot(2:end) + Theta_bar*u_old;
    Ref - Yf_tot(2:end) + Theta_bar*u_old;
    Ref - Yf_tot(2:end) + Theta_bar*u_old;
    -Ref + Yf_tot(2:end) - Theta_bar*u_old;
    -Ref + Yf_tot(2:end) - Theta_bar*u_old;
    -Ref + Yf_tot(2:end) - Theta_bar*u_old;
    -Ref + Yf_tot(2:end) - Theta_bar*u_old;
    -Ref + Yf_tot(2:end) - Theta_bar*u_old;
    DP;]

options = optimset('linprog');
% Set (additional) options:
options = optimset(options,'LargeScale','off','Display','off');

[U, fval] = linprog(f,Aineq,Bineq,[],[],LB,UB,U0,options);
out = [fval; scale'.*U(1:5,1)];

% utot1 = U(1:5:5*(Hp));
% utot2 = U(2:5:5*(Hp));
% utot3 = U(3:5:5*(Hp));
% utot4 = U(4:5:5*(Hp));
% utot5 = U(5:5:5*(Hp));
%
% figure(1)
% subplot(511)
% plot(simtime(1:end-1),utot1,'k-*')
% subplot(512)
% plot(simtime(1:end-1),utot2,'k-*')
% subplot(513)
% plot(simtime(1:end-1),utot3,'k-*')
% subplot(514)
% plot(simtime(1:end-1),utot4,'k-*')
% subplot(515)
% plot(simtime(1:end-1),utot5,'k-*')

end
B.4 2-DOF Model

In the last section of this appendix the equations of motion for the 2-DOF model are given. The derivation is similar to the one done in Chapter 4, but here the lateral velocity of the centre of gravity is taken as an additional state and it is assumed that the longitudinal speed remains constant, to end up with a linear system. Equation (4.12) that describes the influence of steering on the yaw acceleration is then extended with an extra equation [Pac02]

\[
\begin{align*}
    m \left( \dot{v}_y + v_x \dot{\psi} \right) &= b F_{y,lr} + b F_{y,rr} + a F_{y,lf} + a F_{y,rf}, \\
    I_{zz} \ddot{\psi} &= -b F_{y,lr} - b F_{y,rr} + a F_{y,lf} + a F_{y,rf}.
\end{align*}
\]  

(B.1)

Filling in Equation (4.14) and (4.15) for the tyre slip angles, and under the small angle approximation and assumption of small brake forces, this expression can be rewritten to

\[
\begin{align*}
    m \dot{v}_y &= - \left[ m v_x + \frac{2}{v_x} (a C_f - b C_r) \right] \dot{\psi} - \frac{2}{v_x} (C_f + C_r) v_y + 2 C_f \delta \\
    I_{zz} \ddot{\psi} &= - \frac{2}{v_x} \left[ (a C_f - b C_r) \right] v_y - \frac{2}{v_x} \left[ (a^2 C_f + b^2 C_r) \right] \dot{\psi} + 2 a C_f \delta
\end{align*}
\]  

(B.2)

Under the assumption that the brake forces work parallel to the vehicle longitudinal axis, Equation (B.2) can be put in state space format

\[
\begin{align*}
    \dot{x} &= A x + Bu \\
    y &= C x + Du
\end{align*}
\]  

(B.3)

with \( x = (v_y \ \ \psi) \ T \), \( u = [\delta \ T_{b,lf} \ T_{b,lr} \ T_{b,rf} \ T_{b,rr}] \ T \), and

\[
A = \begin{pmatrix}
    -\frac{2}{v_x} (C_f + C_r) & -\frac{1}{m}
    \left[ m v_x + \frac{2}{v_x} (a C_f - b C_r) \right] \\
    -\frac{2}{v_x} \left[ (a C_f - b C_r) \right] & -\frac{2}{v_x} \left[ (a^2 C_f + b^2 C_r) \right]
\end{pmatrix}
\]

\[
B = \begin{pmatrix}
    2 C_f / m & 0 & 0 & 0 & 0 \\
    \frac{2 C_f a}{I_{zz}} & \frac{t_f}{2 I_{zz} \tau_{dyn}} & \frac{t_r}{2 I_{zz} \tau_{dyn}} & \frac{0}{2 I_{zz} \tau_{dyn}} & \frac{0}{2 I_{zz} \tau_{dyn}}
\end{pmatrix}
\]

\[
C = \begin{pmatrix}
    1 & 0 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 & 0
\end{pmatrix}, \quad D = \begin{pmatrix}
    0 & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 0 & 0
\end{pmatrix}.
\]  

71