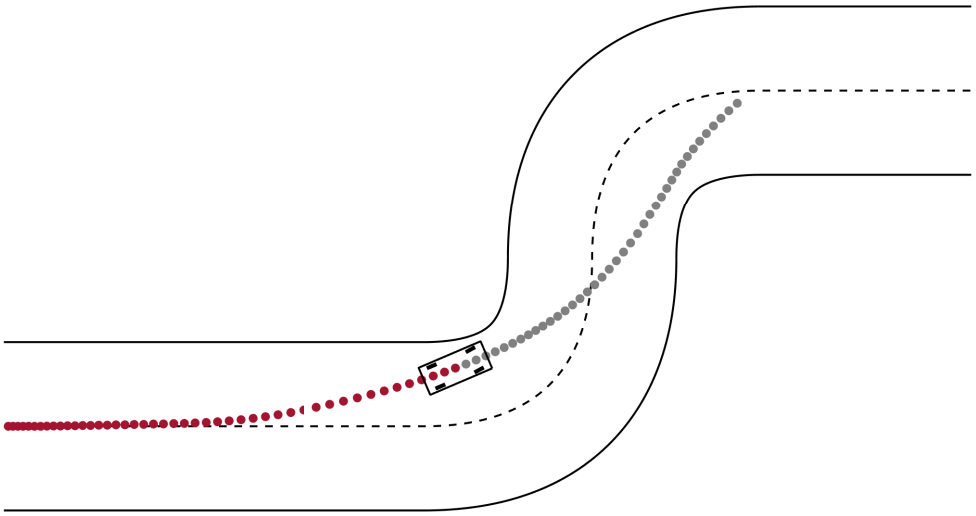
CHALMERS
UNIVERSITY OF TECHNOLOGY



Autonomous high-speed vehicle manoeuvring

An optimal control approach to autonomous driving

Master's thesis in Systems, Control and Mechatronics

Michel Company
Fredrik Andersson

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2019

Master's thesis 2019

Autonomous high-speed vehicle manoeuvring

An optimal control approach to autonomous driving

MICHEL COMPANY
FREDRIK ANDERSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
Gothenburg, Sweden 2019

Autonomous high-speed vehicle manoeuvring
An optimal control approach to autonomous driving
MICHEL COMPANY
FREDRIK ANDERSSON

Supervisor: Balázs Adam Kulcsár, Electrical Engineering
Examiner: Balázs Adam Kulcsár, Electrical Engineering

Master's Thesis 2019
Department of Electrical Engineering
Division of Systems and Control
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2019

Autonomous high-speed vehicle manoeuvring
An optimal control approach to autonomous driving
MICHEL COMPANY
FREDRIK ANDERSSON
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The topic of this thesis is a linear time-varying model predictive contouring control scheme (LTVMPCC) of a nonlinear vehicle model that is linearized around the trajectory. Instead of determining a specific path for a vehicle on a known track the LTVMPCC calculates a control signal in which the vehicle progress furthest for a finite time horizon. The controller can handle physical boundaries of the control inputs such as minimum and maximum steering angles, but also limitations on the states. For example the controller will not allow for a control input in which the position of the vehicle progress outside the track.

An approach to avoid obstacles is also implemented in the controller. As the controller only can handle linear problems a sub-optimal feasible path is found around the obstacles. This path is the fed into the controller by formulating new constraints around the path.

A Lyapunov based approach is taken to give the controller a stabilizing ability. This is done by an addition of a quadratic constraint which forces the weighted norm of the state to decrease. The contraction parameter of the constraint is computed to ensure the stabilizing ability remains despite a model/plant mismatch due to linearization.

As the controller relies on the current states for calculating the control input all the states are required to be known, or at least estimated. This can be achieved by using various sensors combined with an observer. In this thesis an Extended Kalman Filter is used to estimate the states.

The LTVMPCC is tested on different scenarios and compared against a trivial LQ controller.

Keywords: Optimal Control, Vehicle Model, Model Predictive Control, Path Following, Kalman, Stability Guarantee, Lyapunov

Acknowledgements

We would like to extend gratitude to Chalmers University of Technology for letting us realise our idea for this thesis and especially Balázs Kulcsár for his support and tutoring. We could not have asked for a better supervisor.

Michel Company and Fredrik Andersson, Gothenburg, March 2019

Contents

List of Figures

List of Tables

Nomenclature

x	Position global frame
y	Position global frame
φ	Rotation local frame
v_x	Velocity in x-direction local frame
v_y	Velocity in y-direction local frame
ω	Angular velocity local frame
I_z	Inertia around z-axis
a_x	Acceleration in x-direction local frame
a_y	Acceleration in y-direction local frame
m	Mass of the car
l_f	Distance from C.O.G to front axle
l_r	Distance from C.O.G to rear axle
$F_{f,x}$	Longitudinal force, front wheel
$F_{r,x}$	Longitudinal force, rear wheel
$F_{f,y}$	Lateral force, front wheel
$F_{r,y}$	Lateral force, rear wheel
$\mathbf{x}$	Plant state vector, $\mathbf{x} = [x \ \ y \ \ \varphi \ \ v_x \ \ v_y \ \ \omega]^T$
τ	Torque control input
δ	Steering angle control input
$\mathbf{u}$	Plant control input vector, $\mathbf{u} = [\delta \ \ \tau]$
κ	Longitudinal slip ratio
D	Pacejka parameter
C	Pacejka parameter
B	Pacejka parameter
E	Pacejka parameter
α_f	Slip angle, front wheel
α_r	Slip angle, rear wheel
N	Prediction horizon

k	Prediction step
$\mathbf{Q}$	State weighting matrix
$\mathbf{R}$	Control input weighting matrix
$\mathcal{X}$	Set of state constraints
$\mathcal{U}$	Set of control input constraints
e_c	Contour error
e_l	Lag error
T_s	Sample time
v	Virtual control input
θ	Virtual state
$x_d(\theta)$	Path polynomial function, x coordinate
$y_d(\theta)$	Path polynomial function, y coordinate
L_{track}	Path length
q_c	Contour error weight
q_l	Lag error weight
q_θ	Virtual state weight
$\mathbf{w}$	Process noise
$\mathbf{v}$	Observation noise
$\mathbf{P}_f$	Observer covariance matrix
$\hat{\mathbf{x}}$	Estimated state
$\hat{\mathbf{x}}^*$	State trajectory from previous solution, shifted one time step
$\hat{\mathbf{u}}^*$	Control trajectory from previous solution, shifted one time step
$\mathbf{A}_c$	System matrix, continuous time
$\mathbf{B}_c$	System matrix, continuous time
$\mathbf{g}_c$	System matrix, continuous time
$\mathbf{A}_d$	System matrix, discrete time
$\mathbf{B}_d$	System matrix, discrete time
$\mathbf{g}_d$	System matrix, discrete time
$\bar{\mathbf{A}}$	Augmented system matrix
$\bar{\mathbf{B}}$	Augmented system matrix
$\bar{\mathbf{g}}$	Augmented matrix
$\boldsymbol{\xi}$	Controller state vector, $\boldsymbol{\xi} = [x \ \ y \ \ \varphi \ \ v_x \ \ v_y \ \ \omega \ \ \theta]^T$
$\bar{\mathbf{u}}$	Control input vector, $\bar{\mathbf{u}} = [\delta \ \ \tau \ \ v]^T$
$\mathbf{P}$	Contraction constraint weighting matrix
ϵ	Slack variable

$\mathbf{R}_\Delta$	Weighting matrix for control input differences
$\hat{\mathbf{\Theta}}^*$	Theta trajectory from previous solution, shifted one time step
$\hat{\xi}^*$	Controller state trajectory from previous solution, shifted one time step
$\mathbf{H}$	Hessian matrix of QP problem
$\mathbf{f}$	Linear part of QP problem
$\mathbf{A}_{eq}$	A matrix, equality constraint
$\mathbf{b}_{eq}$	b matrix, equality constraint
$\mathbf{A}_{in}$	A matrix, inequality constraint
$\mathbf{b}_{in}$	b matrix, inequality constraint
$\mathbf{Q}_{qc}$	Q matrix, quadratic constraint
$\mathbf{l}_{qc}$	Linear part, quadratic constraint
r_{qc}	Right hand side, quadratic constraint
$\mathbf{z}$	Optimisation variable vector
$\dot{\mathbf{r}}$	Observer state

1
Introduction

Autonomous driving is an important area of research as many car-manufactures are working hard to deliver self-driving cars in the near future. These cars needs to be controlled by a computer, where the steering angle, throttle- and brake input needs to be momentarily decided by some algorithm. There are many different suitable algorithms, simple PID algorithms to more advanced algorithms such as Model Predictive Control. The latter uses a mathematical model to predict "what's going to happen". Much like humans use our senses to take corrective actions for example when driving.
Several works on autonomous driving using Model Predictive Control exists. In [8] MPC is used to perform a double lane change manoeuvre in "snowy" conditions while in [9] the work is focused on autonomous driving at the limits of handling capabilities.
However, there are a distinct lack of works on autonomous driving which combine detailed analysis of the controller, especially in different scenarios, and stability guarantees.

1.1 Aim of the project

The aim of this thesis is to investigate how optimal control theory can be used to improve vehicle manoeuvring in the context of autonomous high-speed driving. One example of high-speed driving would be racing, another example would be high-speed evasive manoeuvres from obstacles on the road. Both will be investigated in this thesis. This includes developing both a control algorithm and a path planner algorithm, which both should be suitable for real time implementation.
In contrast to similar works, this will take a more scientific approach and analyse the performance of the controller in detail, and in various scenarios. An approach to stability guarantees of the control algorithm based on the theories presented in [2] will also be introduced. This thesis will present an approach to implementation in the control algorithm of the theories on Linear Time Varying systems which is not present in other works.

1.2 Limitations

This work will focus solely on high-speed driving at the limits of handling as there are already many works on autonomous driving in within reasonable limits, such as [8]. The work will use methods and algorithms suitable for real-time implementation

since as stated the goal is to develop something that could eventually be implemented in a real car. Thus the use more advanced and computationally heavy algorithms for both path planning and control have been discarded. The exception to this is the baseline controller which is only used for comparison purposes. To limit the scope, four scenarios has been chosen for evaluation of the control algorithms. They include, high-speed cornering, obstacle avoidance, double lane change manoeuvre as well as driving around a race track.

1.3 Disposition

The thesis is structured to give a brief theoretical background, show how the theories have been used and implemented and finally results will be presented and discussed.

Chapter 2 Introduces related theory used in this thesis, including vehicle model, Model Predictive Control basics and Observer theory.

Chapter 3 Describes the methods used for modelling and developing the control algorithms.

Chapter 4 Presents a concept of how to implement this work on hardware. This includes implementation on a small-scale Radio Controlled car using low-level programming language and setting up necessary hardware and software.

Chapter 5 Presents and analyses the results from simulation, divided into four sections, one for each scenario which will be evaluated.

Chapter 6 Analyses the work and presents thoughts and ideas of what to improve.

2
Theory

The following chapter will give a theoretical background of the methods used in this thesis, such that it will be easy to follow the procedures in Chapter 3. The dynamical model will be presented along with the so called Pacejka magic formula. A brief background on Model Predictive Control (MPC) is given and then the introduction of the Model Predictive Contouring Control (MPCC) scheme follows. Finally a general description of the observer is given.

2.1 Vehicle model

An overall view of different models describing the dynamics of a vehicle can be found here [6]. Below will focus on one of the most commonly used model, namely the single-track vehicle model (also known as the bicycle model). It is also the one used in this thesis. See Figure 2.1 for a visual overview of the model.

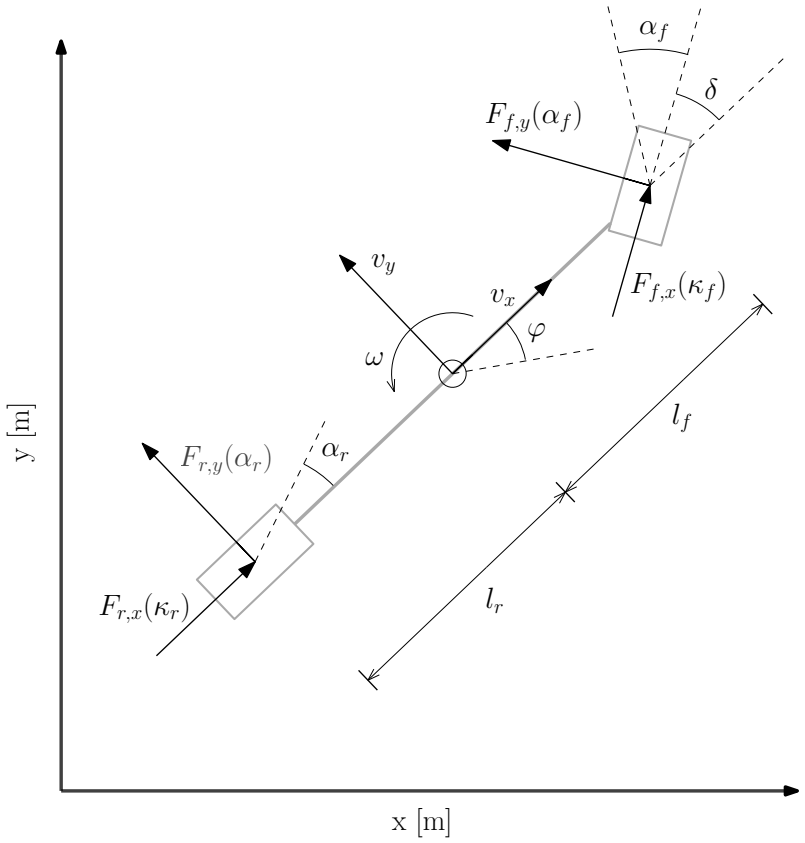


Figure 2.1: Single track vehicle model.

The single track model a simplified vehicle model where symmetry of the vehicle is assumed. Movement in the x-y plane is only considered, i.e no pitch or roll movements are considered. And lastly vertical load is taken as constant. The set of equations which describe the vehicle motion is as follows:

$$\dot{x} = v_x \cos\varphi - v_y \sin\varphi \tag{2.1a}$$
$$\dot{y} = v_x \sin\varphi + v_y \cos\varphi \tag{2.1b}$$
$$\dot{\varphi} = \omega \tag{2.1c}$$
$$\dot{v}_x = \frac{1}{m}(F_{r,x} + F_{f,x}\cos\delta - F_{f,y}\sin\delta + mv_y w) \tag{2.1d}$$
$$\dot{v}_y = \frac{1}{m}(F_{r,y} + F_{r,x}\sin\delta + F_{f,y}\cos\delta - mv_x w) \tag{2.1e}$$
$$\dot{\omega} = \frac{1}{I_z}(l_f F_{f,y}\cos\delta - l_r F_{r,y}) \tag{2.1f}$$
$$\dot{\mathbf{x}} = [\dot{x} \quad \dot{y} \quad \dot{\varphi} \quad \dot{v}_x \quad \dot{v}_y \quad \dot{\omega}]^T \tag{2.1g}$$
$$\mathbf{u} = [\delta \quad \tau]^T \tag{2.1h}$$

Where m is the mass, I_z is the inertia around the z-axis, l_r, l_f is the length to the rear- and front axis respectively from the centre point .The three first equations are kinematic equations describing the vehicle motion with respect to a global frame. The last three equations are dynamic equations which describes the vehicle motion

with respect to a fixed body frame. The input to the model is defined by 2.1h and consists of the steering angle δ and torque input τ.
There are several ways to model the forces which acts on the tyres. A good overview of these models such as the dynamic LuGre model, the semi-empirical TMEasy model and the well known Pacejka magical formula can be found here [7]. Using the latter of the models, the longitudinal force at the fron and rear wheel is given by:

$$F_{i,x} = D_i \sin\left(C_i \arctan\left(B_i\kappa_i - E_i(B_i\kappa_i - \arctan\left(B_i\kappa_i\right))\right)\right),\ i = f, r \tag{2.2}$$

Where κ is the longitudinal slip ratio, defined as:

$$\kappa_i = \frac{\omega_w^i r_w^i - v_x}{v_x},\ i = f, r \tag{2.3}$$

Where r_w is the radius of the wheel. The slip ratio is measure of the difference in the rotational speed of the wheel ω_w and longitudinal velocity of the vehicle. While the lateral force is then given by:

$$F_{i,x} = D_i \sin\left(C_i \arctan\left(B_i\alpha_i - E_i(B_i\alpha_i - \arctan\left(B_i\alpha_i\right))\right)\right),\ i = f, r \tag{2.4}$$

Where α_i is the slip angles, defined as:

$$\alpha_f = -\arctan\left(\frac{\omega l_f + v_y}{v_x}\right) + \delta \tag{2.5}$$

$$\alpha_r = \arctan\left(\frac{\omega l_r - v_y}{v_x}\right) \tag{2.6}$$

The slip angle is a measure of the difference in the angle of the velocity vector of the wheel and the heading of the wheel.
The Pacejka magical formula is a semi-empirical formula, i.e its equations are not derived from any physical laws hence the parameters $D_i, C_i, B_i, E_i,\ i = f, r$ needs to be decided from an identification process.

2.2 Model Predictive Control

Model predictive control (MPC) has been used in industry since the 1980's. Mainly in the process industry where sampling time is in the magnitude of minutes, as the computational requirements of MPC is quite heavy. This method uses a mathematical model to predict future behaviour, see figure 2.2. In contrast with the past there is now embedded hardware which can handle MPC even at low sampling times. Given this development in hardware, and the fact that MPC is the method which mimics human-like anticipation the best, it is now suited for embedded control in applications like autonomous driving.
Model Predictive control works by solving a finite horizon optimisation problem at each time step:

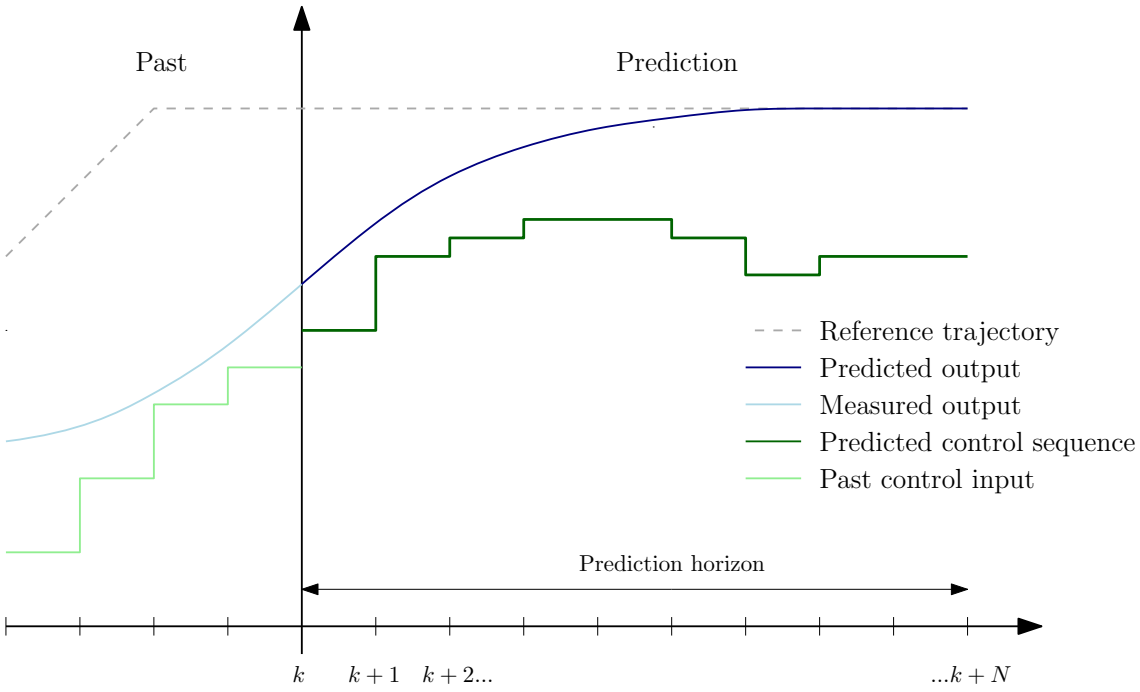


Figure 2.2: Model Predictive Control scheme.

$$\min_{\mathbf{u},\boldsymbol{x}} \quad \sum_{k=0}^{N-1} ||\boldsymbol{x}_k - \boldsymbol{x}_k^{ref}||_{\mathbf{Q}_{mpc}} + ||\boldsymbol{u}_k - \boldsymbol{u}_k^{ref}||_{\mathbf{R}_{mpc}} \tag{2.7a}$$

$$\text{s.t.} \quad \boldsymbol{x}_0 = \boldsymbol{x}(t) \tag{2.7b}$$

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad k = 0, \ldots, N-1 \tag{2.7c}$$

$$\boldsymbol{x}_k \in \mathcal{X}, \quad k = 1, \ldots, N \tag{2.7d}$$

$$\boldsymbol{u}_k \in \mathcal{U}, \quad k = 0, \ldots, N-1 \tag{2.7e}$$

Where k is the discrete time notation with the time step T_s, N is the horizon length, $\boldsymbol{x}_k^{ref}, \mathbf{u}_k^{ref}$ is the reference trajectories (these can be set to zero if the goal is to regulate the system to the origin) and $\mathbf{Q}_{mpc}, \mathbf{R}_{mpc}$ are weighting matrices. The above optimisation problem is stated using the standard quadratic cost function, but other objectives can be added to the cost as well.

2.3 Model Predictive Contouring Control

An adaptation of the standard Model Predictive Control scheme is the Model Predictive Contouring Control scheme [1]. This is a path following scheme, in contrast to the standard reference tracking scheme. The difference between trajectory tracking and path following is that the former explicitly defines where to be and when to be there while the latter only defines where to be. The timing of when to be where on the path is given by a timing law who's evolution is a part of the optimisation problem. This means that the only information needed a priori is the geometric path to be followed and a upper bound on the timing law input. In terms of autonomous

driving this is readily available information, the path to be followed can simply be the fictional line in between the lines that defines the lane and the upper bound on the timing law is the currently maximum allowed or desired velocity. In the Model Predictive Contour Control scheme, the path is called the contour and the contour error is the deviation from the path along its normal vector. Figure 2.3 shows a visualisation of MPCC:

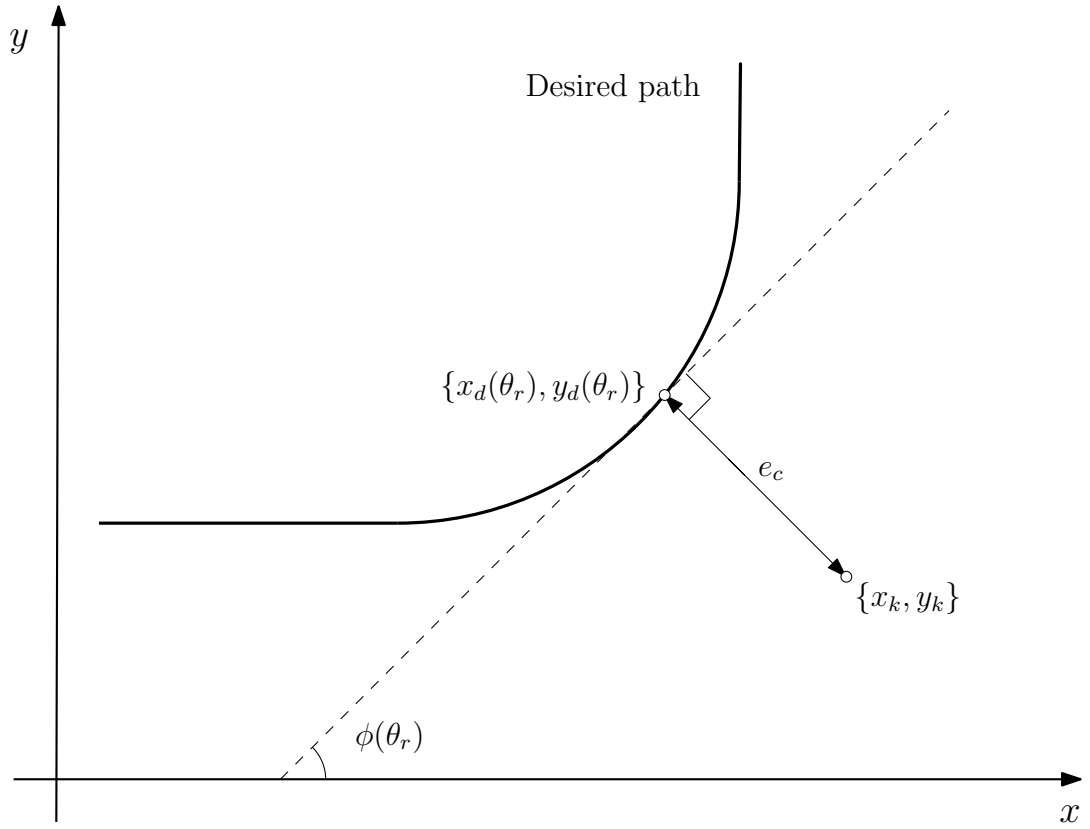


Figure 2.3: Visualisation of the Model Predictive Contouring Control scheme.

Where the contour error e_k^c is given by:

$$e_k^c = \sin\phi(\theta_r)(x_k - x_d(\theta_r)) - \cos\phi(\theta_r)(y_k - y_d(\theta_r)) \tag{2.8}$$

and $\phi(\theta_r)$ is given by:

$$\phi(\theta_r) = \arctan\frac{\nabla y_d(\theta_r)}{\nabla x_d(\theta_r)} \tag{2.9}$$

Where $\{x_d(\theta), y_d(\theta)\}$ is the x and y coordinates given by path polynomials as function of θ. The parameter $\theta_r(x, y)$ is the path parameter value at the position where the contour error is minimised. Finding $\theta_r(x, y)$ becomes an optimisation problem itself, thus it is not suited for Model Predictive Control. In [1] it is proposed to use θ_k as an approximation to θ_r, where the former is given by:

$$\theta_{k+1} = \theta_k + T_s\, v_k, \;\; v_k \in [0, v_{max}], \;\; v_{max} > 0 \tag{2.10}$$

Where v_k is a virtual input, to be decided by the controller. This is the so called timing law often found in path following schemes. In order to make the approximation valid, a measure of the path distance between θ_k and θ_r, called lag error e_l is introduced. Then e_l can be approximated by $\hat{e}_l$. Under the assumption that θ_k is close to $\theta_r(x, y)$, then $\hat{e}_l$ is given by:

$$\hat{e}_k^l = -\cos\phi(\theta_k)(x_k - x_d(\theta_k)) - \sin\phi(\theta_k)(y_k - y_d(\theta_k)) \tag{2.11}$$

And now, using θ_k, an approximation of the contour error can be made:

$$\hat{e}_k^c = \sin\phi(\theta_k)(x_k - x_d(\theta_k)) - \cos\phi(\theta_k)(y_k - y_d(\theta_k)) \tag{2.12}$$

Figure 2.4 shows the visualisation of these two equations. As can be seen when $\hat{e}_l \to 0$ then $\theta_k \to \theta_r$ thus making $\hat{e}_c$ a valid approximation of e_c.

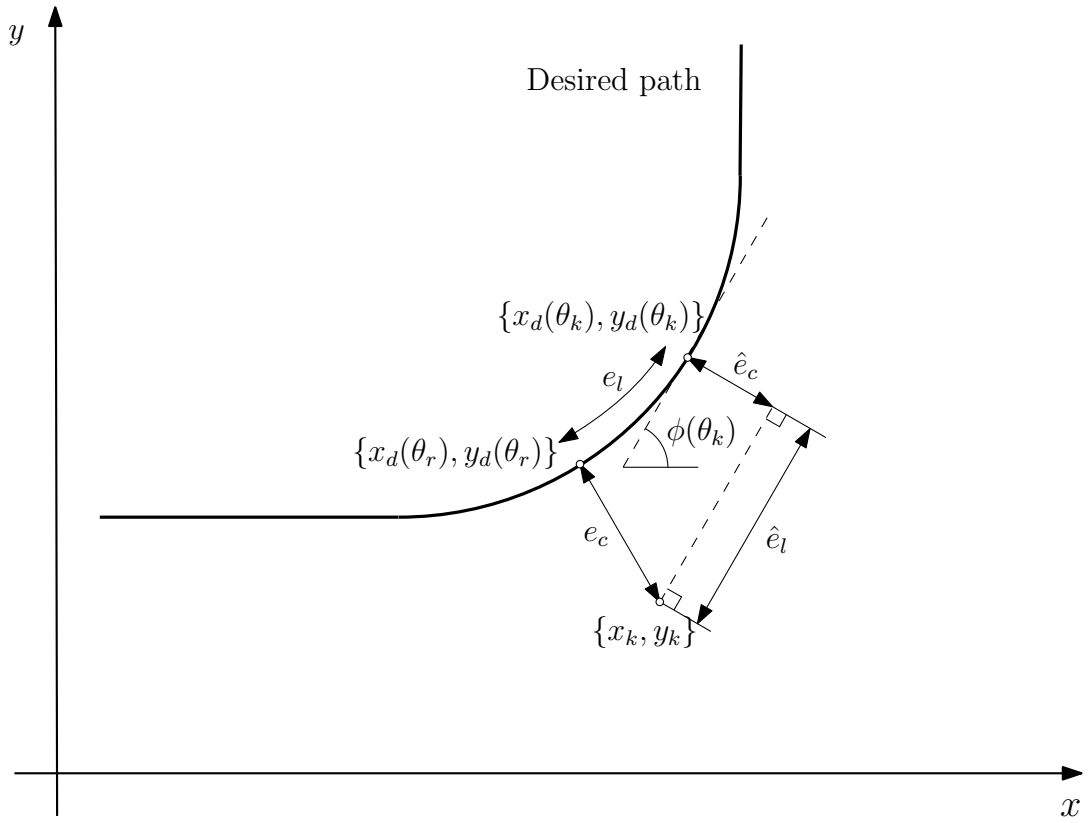


Figure 2.4: Visualisation of the approximate Model Predictive Contouring Control scheme.

The resulting optimisation problem formulation thus becomes:

$$\min_{\boldsymbol{x},\boldsymbol{\theta}} \quad \sum_{k=0}^{N-1} ||\hat{e}_k^c(x_k, y_k, \theta_k)||_{q_c} + ||\hat{e}_k^l(x_k, y_k, \theta_k)||_{q_l} - q_\theta \theta_k \tag{2.13a}$$

$$\text{s.t.} \quad \boldsymbol{x}_0 = \boldsymbol{x}(t) \tag{2.13b}$$

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k), \quad k = 0, \ldots, N-1 \tag{2.13c}$$

$$\theta_{k+1} = \theta_k + T_s\, v_k, \quad k = 0, \ldots, N-1 \tag{2.13d}$$

$$\boldsymbol{x}_k \in \mathcal{X}_k, \quad k = 1, \ldots, N \tag{2.13e}$$

$$\theta_k \in [-L_{track}, 0], \quad k = 1, \ldots, N \tag{2.13f}$$

$$\boldsymbol{u}_k \in \mathcal{U}, \quad k = 0, \ldots, N-1 \tag{2.13g}$$

$$v_k \in [0, v_{max}], \quad k = 0, \ldots, N-1 \tag{2.13h}$$

Where q_c, q_l, $q_\theta \geq 0$ and $v_{max} > 0$.

Mostly used in industry for machining tools or robotic arms, Model Predictive Contouring Control has in fact been successfully used for autonomous racing by a team at ETH, where both linear [3] and nonlinear [10] versions have been implemented.

2.4 Observer

In order to implement the control scheme described in 2.3 the current state of the system is required to be known, or at least estimated. This is achieved by fusing data from the system model, measured data and the control inputs. The method used in this thesis is the Extended Kalman filter. The system model and the observation model can generally be described in discrete time as:

$$\begin{aligned} \mathbf{x}_k &= \mathbf{f}(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \\ \mathbf{z}_k &= \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k \end{aligned} \tag{2.14a}$$

Where $\mathbf{w}_k$ and $\mathbf{v}_k$ are asumed to be:

$$\begin{aligned} \mathbf{w}_k &\sim \mathcal{N}(\mathbf{0}, {\sigma_\mathbf{w}}^2), \qquad \sigma_\mathbf{w} = \boldsymbol{Q}_k \\ \mathbf{v}_k &\sim \mathcal{N}(\mathbf{0}, {\sigma_\mathbf{v}}^2), \qquad \sigma_\mathbf{v} = \boldsymbol{R}_k \end{aligned} \tag{2.15}$$

The filter then works by using a model to first calculate a prediction of the state vector, $\hat{\mathbf{x}}$ and the covariance matrix $\boldsymbol{P}_f$:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{f}(\hat{\mathbf{x}}_{k|k-1}, \mathbf{u}_k) \tag{2.16a}$$

$$\boldsymbol{P}^f_{k|k-1} = \mathbf{F_k} \boldsymbol{P}^f_{k-1|k-1} \mathbf{F}_k^T + \boldsymbol{Q}_k \tag{2.16b}$$

The second step is to include the observations from the current sample and update the state vector and the covariance matrix, this is done by first calculating the new measurement residual $\tilde{\mathbf{y}}_k$ followed by the innovation covariance $\mathbf{S}_k$ and the Kalman gain $\mathbf{K}_k$:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{h}(\hat{\mathbf{x}}_{k|k-1}) \tag{2.17a}$$
$$\mathbf{S}_k = \boldsymbol{H}_k \boldsymbol{P}^f_{k|k-1} \boldsymbol{H}^T_k + \boldsymbol{R}_k \tag{2.17b}$$
$$\mathbf{K}_k = \boldsymbol{P}^f_{k|k-1} + \boldsymbol{H}^T_k \mathbf{S}^{-1}_k \tag{2.17c}$$
$$\hat{\mathbf{x}} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \tag{2.17d}$$
$$\boldsymbol{P}^f_{k|k} = (\mathbf{I} - \mathbf{K}_k \boldsymbol{H}_k) \boldsymbol{P}^f_{k|k-1} \tag{2.17e}$$

where the state transition and observation matrices are calculated from the following Jacobians:

$$\begin{aligned} \mathbf{F}_k &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{u}_k} \\ \boldsymbol{H}_k &= \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}_{k|k-1}} \end{aligned} \tag{2.18}$$

3
Methods

The methods used in this thesis will be presented in the following chapter. The dynamical model used will be presented, along with corresponding parameter values. Linearization procedure of the optimal control problem will be shown and the controller constraints will be derived as well as the conversion of the optimal control problem to a convex Quadratically Constrained Quadratic Programming problem. The chapter will also introduce a Baseline controller which will be used later as a benchmark. Lastly a section on the method of state estimation used in the thesis is given.

3.1 Vehicle model

The vehicle model used in this thesis is the one described in section 2.1 with parameters based on the Kyosho Mini-Z SPORTS AWD radio controlled car. The lateral forces acting on the wheels are modelled by the simplified Pacejka magic formula. By taking 2.4 and setting $E_i = 0$ the lateral forces are given by:

$$F_{i,x} = D_i \sin\left(C_i \arctan\left(B_i \alpha_i\right)\right), \quad i = f, r \tag{3.1}$$

The Pacejka formula is not used to model the longitudinal forces as they are function of the slip ratio 2.3 which requires the wheel speed to be measurable. That is not possible with the current hardware setup so the longitudinal forces acting on the wheels are instead modelled by:

$$F_{i,x} = \frac{\tau}{r_w} - 0.5 F_{resist}, \quad i = f, r \tag{3.2}$$

Where τ is the input torque, r_w is the radius of the wheel and F_{resist} is the combined air resistance force and rolling resistance force. Assuming zero slope, the resistance force can be modelled as:

$$F_{resist} = C_{rr} m g + 0.5 A C_{air} v_x^2 \rho \tag{3.3}$$

Where C_{rr} is the rolling resistance coefficient, A is the projected front area of the car and ρ is the density of air. C_{rr} is taken as the standard value for a car with rubber tires on asphalt concrete.

The complete list of parameters for the car is then as follows:

Vehicle parameter values	
Parameter	Value
m	$0.189\,kg$
l_f	$0.047\,m$
l_r	$0.047\,m$
r	$0.0125\,m$
I_z	$0.303975 \cdot 10^{-3}$
A	$0.0035\,m^2$
w_{car}	0.07
l_{car}	0.12
C_{rr}	0.01
C_{air}	0.26
B_f	0.3
B_r	0.3
C_f	1.3
C_r	1.3
D_f	4
D_r	4

Table 3.1: Table of vehicle parameter values.

3.2 Linear Time Varying Model Predictive Contouring Control

In order to solve the optimisation problem stated in section 2.3 sufficiently fast we need to convert it into a linear problem with quadratic and/or linear cost function. Since the model used is nonlinear we must linearise it. Let $\dot{\boldsymbol{x}} = \mathbf{f}(\boldsymbol{x}, \mathbf{u})$ and $\boldsymbol{x} = [x \;\; y \;\; \phi \;\; v_x \;\; v_y \;\; w]^T$, also let $\hat{\boldsymbol{x}}^*$ and $\hat{\boldsymbol{u}}^*$ be the optimal trajectories and the optimal control inputs from the previous solution shifted one time step, then the linearized model is given by:

$$\begin{gathered}\dot{\boldsymbol{x}} \approx \mathbf{f}(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*) + \nabla \mathbf{f}(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*) \begin{bmatrix} \boldsymbol{x} - \hat{\boldsymbol{x}}^* \\ \mathbf{u} - \hat{\mathbf{u}}^* \end{bmatrix} = \\ \mathbf{f}(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*) + \mathbf{A}_c(\boldsymbol{x} - \hat{\boldsymbol{x}}^*) + \mathbf{B}_c(\mathbf{u} - \hat{\mathbf{u}}^*) = \mathbf{A}_c\boldsymbol{x} + \mathbf{B}_c\mathbf{u} + \mathbf{g}_c\end{gathered} \tag{3.4}$$

Where $\mathbf{A}_c, \mathbf{B}_c$ and $\mathbf{g}_c$ are given by:

$$\begin{aligned}\mathbf{A}_c &= \left.\frac{\partial \mathbf{f}}{\partial \boldsymbol{x}}\right|_{(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*)} \\ \mathbf{B}_c &= \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*)} \\ \mathbf{g}_c &= f(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*) - \mathbf{A}_c\hat{\boldsymbol{x}}^* - \mathbf{B}_c\hat{\mathbf{u}}^*\end{aligned}$$

The optimal shifted input trajectory is calculated by truncating the previous solution and appending a feasible input:

$$\hat{\mathbf{u}}^* = \{\mathbf{u}^*_{k|k-1}, \mathbf{u}^*_{k+1|k-1} \cdots \quad \cdots \mathbf{u}^*_{k+N-2|k-1,} \hat{\mathbf{u}}^*_{k+N-1}\} \tag{3.5}$$

Where the appended input $\hat{\mathbf{u}}^*_{k+N-1} = \mathbf{u}^*_{k+N-2|k-1}$.
The optimal shifted state trajectory is calculated by truncating the previous solution and appending a feasible state:

$$\hat{\boldsymbol{x}}^* = \{\boldsymbol{x}_k, \boldsymbol{x}^*_{k+1|k-1} \cdots \quad \cdots \boldsymbol{x}^*_{k+N-1|k-1}, \hat{\boldsymbol{x}}^*_{k+N}\} \tag{3.6}$$

Where the appended state $\hat{\boldsymbol{x}}^*_{k+N} = \mathbf{f}(\boldsymbol{x}^*_{k+N-1|k-1}, \hat{\mathbf{u}}^*_{k+N-1})$. This is the Jacobian linearization method where the nominal trajectories are contained in $\mathbf{g}_c$.
The system matrices are then discretized using the zero-order hold method:

$$\mathbf{M} = \begin{bmatrix} \mathbf{A}_c & \mathbf{B}_c & \mathbf{g}_c \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \qquad e^{\mathbf{M}} = \begin{bmatrix} \mathbf{A}_d & \mathbf{B}_d & \mathbf{g}_d \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{3.7}$$

The error measurement equations e_c, e_l must also be linearised. As they are functions of $x_d(\theta_k), y_d(\theta_k)$, the path functions must first be linearised. This is done by Taylor expanding the path functions around $\hat{\boldsymbol{\Theta}}^*$ and neglecting higher order terms. Where $\hat{\boldsymbol{\Theta}}^*$ is the optimal trajectory of $\boldsymbol{\theta}$ from the previous solution shifted one time step, and appended in same manner as 3.6. The path function $\hat{x}^d_{k+1}(\theta_k, \hat{\Theta}^*_k)$ is given by:

$$\hat{x}^d_{k+1}(\theta_k, \hat{\Theta}^*_k) = x_d(\hat{\Theta}^*_{k+i|k}) + \nabla x_d(\hat{\Theta}^*_{k+i|k})(\theta_k - \hat{\Theta}^*_{k+i|k}) \tag{3.8}$$

The path function $\hat{y}^d_{k+1}(\theta, \hat{\Theta}^*_k)$ is given by:

$$\hat{y}^d_{k+1}(\theta_k, \hat{\Theta}^*_k) = y_d(\hat{\Theta}^*_{k+i|k}) + \nabla y_d(\hat{\Theta}^*_{k+i|k})(\theta_k - \hat{\Theta}^*_{k+i|k}) \tag{3.9}$$

Now the contour error and the lag error equations can be given by their linear approximations:

$$\hat{e}^l_{k+i} = -\cos\phi(\hat{\theta}^*_{k+i|k})(x_k - \hat{x}^d_{k+1}(\theta_k, \hat{\Theta}^*_k)) - \sin\phi(\hat{\theta}^*_{k+i|k})(y_k - \hat{y}^d_{k+1}(\theta_k, \hat{\Theta}^*_k)) \tag{3.10}$$

$$\hat{e}^c_{k+i} = \sin\phi(\hat{\theta}^*_{k+i|k})(x_k - \hat{x}^d_{k+1}(\theta_k, \hat{\Theta}^*_k)) - \cos\phi(\hat{\theta}^*_{k+i|k})(y_k - \hat{y}^d_{k+1}(\theta_k, \hat{\Theta}^*_k)) \tag{3.11}$$

The state vector is now augmented with the dynamics of θ, from eq. 2.10, yielding the controller state vector $\boldsymbol{\xi} = [x \;\; y \;\; \varphi \;\; v_x \;\; v_y \;\; w \;\; \theta]^T$ and the controller input vector $\bar{\boldsymbol{u}} = [\delta \;\; \tau \;\; v_k]^T$. The linearized system matrices are augmented as following:

$$\bar{\mathbf{A}} = \begin{bmatrix} \mathbf{A}_d & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}, \quad \bar{\mathbf{B}} = \begin{bmatrix} \mathbf{B}_d & \mathbf{0} \\ \mathbf{0} & T_s \end{bmatrix}, \quad \bar{\mathbf{g}} = \begin{bmatrix} \mathbf{g}_d \\ \mathbf{0} \end{bmatrix} \tag{3.12}$$

Using the above results, the linearized optimisation problem can be formulated:

$$\begin{aligned}
\min_{\bar{\mathbf{u}},\boldsymbol{\xi},\boldsymbol{\epsilon}} \quad & \sum_{k=0}^{N-1} ||\hat{e}_k^c(x_k,y_k,\theta_k)||_{q_c} + ||\hat{e}_k^l(x_k,y_k,\theta_k)||_{q_l} - q_\theta\theta_k + && \text{(3.13a)}\\
& ||\Delta\bar{\boldsymbol{u}}_k||_{\boldsymbol{R}_\Delta} + ||\boldsymbol{\epsilon}||_{q_{s^2}} + \boldsymbol{\epsilon}_{q_s} && \\
\text{s.t.} \quad & \boldsymbol{\xi}_0 = \boldsymbol{\xi}(t) && \text{(3.13b)}\\
& \boldsymbol{\xi}_{k+1} = \bar{\boldsymbol{A}}_k\boldsymbol{\xi}_k + \bar{\boldsymbol{B}}_k\bar{\boldsymbol{u}}_k + \bar{\boldsymbol{g}}_k, \quad k = 0,\dots,N-1 && \text{(3.13c)}\\
& \boldsymbol{\xi}_k \in \mathcal{X}_k, \quad k = 1,\dots,N && \text{(3.13d)}\\
& \bar{\boldsymbol{u}}_k \in \mathcal{U}, \quad k = 0,\dots,N-1 && \text{(3.13e)}\\
& \epsilon_k \in [0,\infty], \quad k = 0,\dots,N-1 && \text{(3.13f)}\\
& ||\boldsymbol{\xi}^{j+1}||_{\mathbf{P}} \leq \alpha||\boldsymbol{\xi}^j||_{\mathbf{P}} && \text{(3.13g)}
\end{aligned}$$

Both state and input are subject to constraints. A lower bound on the state v_x is imposed as the model becomes numerically unstable at longitudinal velocities close to zero. This is a well known issue with the single-track vehicle model. The lateral velocity v_y is also symmetrically bounded at ± 3 m/s. Input are subject to upper and lower bounds:

$$-\frac{\pi}{5}\,rad \;\leq\; \delta \;\leq\; \frac{\pi}{5}\,rad \tag{3.14a}$$

$$-0.003\,Nm \;\leq\; \tau \;\leq\; 0.003\,Nm \tag{3.14b}$$

And the virtual input v_k is upper bounded by v_{max}. The matrix $\boldsymbol{R}_\Delta$ is the penalty for deviations of the control input and is given by $\boldsymbol{R}_\Delta = diag(r_\delta, r_\tau, r_v)$. The last constraint 3.13g is the so called contraction constraint and is included to stabilize the closed loop.
For Model Predictive Contouring Control the path must be differentiable. In this thesis the points are interpolated using third degree parametric polynomials. Where the parameter is $\theta_k \in [-L_{track}, 0]$, $\theta \leq 0$, and where L_{track} is the total length of the centre line. Note that the centre line is negatively parameterzsed, i.e it ends in the origin. This is done for stability purposes, explained in more detail in section 3.3.4. It is preferable to keep the ratio between q_c and q_l to simplify the tuning of parameters which makes $\hat{e}_l \to 0$. Thus q_θ is considered as the performance tuning weight. Figure 3.1 shows the effects of the solution for different values of q_θ.

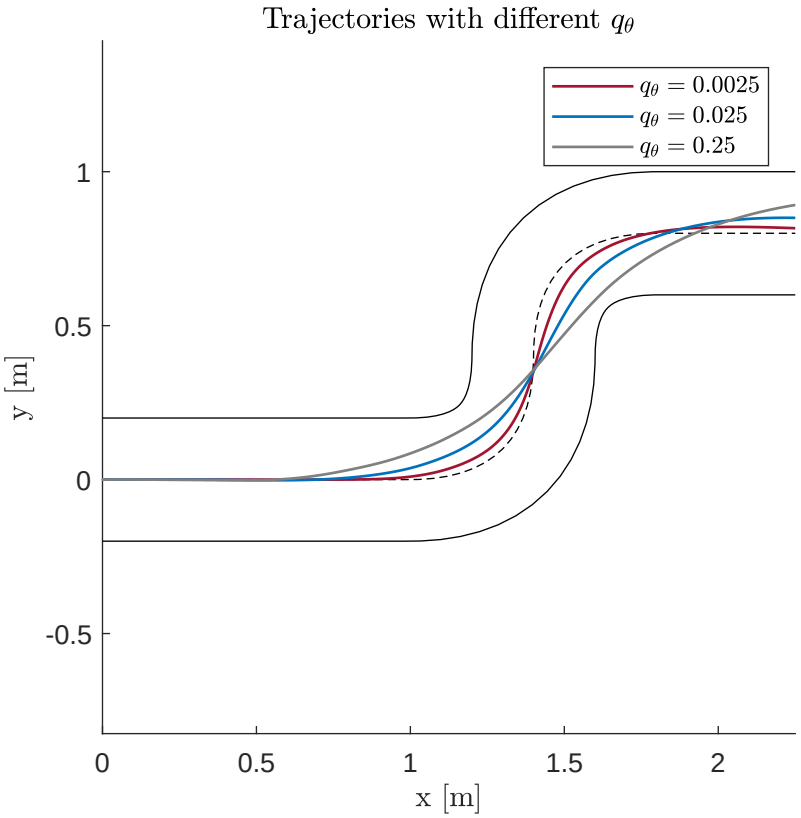


Figure 3.1: Driven trajectories for different values of q_θ.

Adapted from [1], for $k = 0$ the following procedure is used to find initial state and input trajectories to linearise around:

Step 1: Initialise $\hat{x}_0^{*0}$, $\hat{u}_0^{*0}$, $\hat{\Theta}_k^{*0}$ to $\hat{x}^* = \{x_0, x_0.....x_0, x_0\}$, $\hat{u}^* = \{u_0, u_0.....u_0, u_0\}$, $\hat{\Theta}^* = \{\theta_0, \theta_0.....\theta_0, \theta_0\}$ and set $l = 0$
Step 2: Compute 3.13c using $\hat{\xi}_0^{*l}$ (where $\hat{\xi}_0^{*l}$ is $\hat{x}_0^{*l}$ augmented with $\hat{\Theta}_k^{*l}$) and $\hat{u}_0^{*l}$, compute 3.10, 3.11 using $\hat{\Theta}_0^{*l}$
Step 3: Solve optimisation problem 3.13b, increment l and compute $\hat{x}_0^{*l}$, $\hat{u}_0^{*l}$,$\hat{\Theta}_0^{*l}$
Step 4: Repeat Steps 2-4 until $||\hat{\Theta}_0^{*l} - \hat{\Theta}_0^{*l-1}|| \leq \epsilon$ for some $\epsilon > 0$

The linear time varying model predictive contouring control scheme can be summarised as follows:
Step 1: Set $j = 0$, $k = 0$ and use the initial trajectory method to calculate $\hat{x}_0^*$,$\hat{u}_0^*$,$\hat{\Theta}_0^*$
Step 2: Compute 3.13c using $\hat{\xi}_k^*$ and $\hat{u}_k^*$, compute 3.10, 3.11 using $\hat{\Theta}_k^*$
Step 3: Solve optimisation problem 3.13b to obtain u^*, v^*.
Step 4: Apply $u^*(:, 1)$ to the plant, and $v^*(1)$ to 2.10.
Step 5: Calculate $\hat{x}_{k+1}^*$,$\hat{u}_{k+1}^*$,$\hat{\Theta}_{k+1}^*$
Step 6: If $k = jN + N - 1$ then increment j and update 3.13g. Then increment k and return to Step 2.

3.3 Constraints

Besides the state and input constraints detailed above, some additional constraints have been introduced to the controller. They will be described in detail in this section.

3.3.1 Border constraint

In order to stay within the lane two linear half plane constraints are constructed by taking the tangent line at the position of θ_k on the centre line and translating it along the normal. Figure 3.2 shows a visual representation of how the two constraints are constructed:

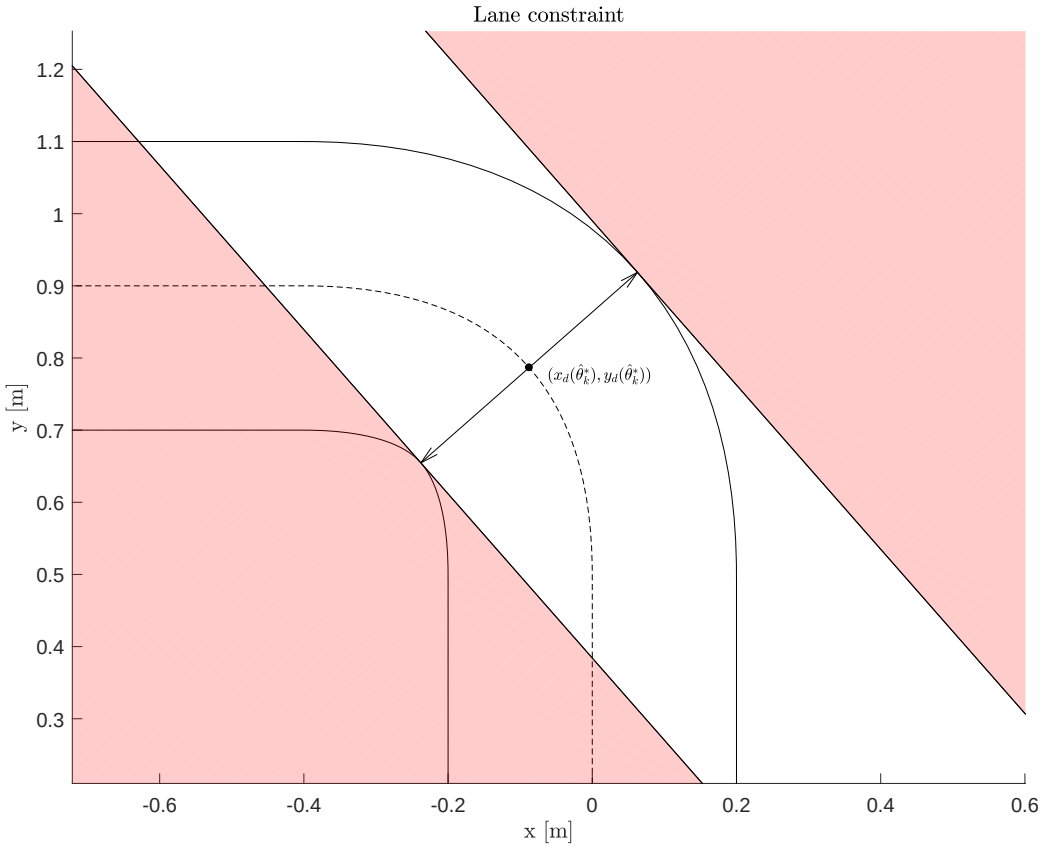


Figure 3.2: Visualisation of lane constraint.

Where the shaded red areas are the forbidden areas.

Let $x_d(\theta), y_d(\theta)$ be the x,y coordinates of θ_k, and w be the width of the lane, then the tangent vector at the position of θ_k is given by:

$$\mathbf{t} = [\cos(\phi(\theta_k)) \quad \sin(\phi(\theta_k))]^T \tag{3.15}$$

And the normal vector at the position of θ_k is given by:

$$\mathbf{n} = [\sin(\phi(\theta_k)) \quad -\cos(\phi(\theta_k))]^T \tag{3.16}$$

This yields the two constraints:

$$\begin{aligned} -(x_k - (x_d(\theta_k) + \frac{w}{2}\sin(\phi(\theta_k)))\sin(\phi(\theta_k)) + \\ +(y_k - (y_d(\theta_k) - \frac{w}{2}\cos(\phi(\theta_k)))\cos(\phi(\theta_k)) \geq 0) \end{aligned} \tag{3.17}$$

$$\begin{aligned} -(x_k - (x_d(\theta_k) - \frac{w}{2}\sin(\phi(\theta_k)))\sin(\phi(\theta_k)) + \\ +(y_k - (y_d(\theta_k) + \frac{w}{2}\cos(\phi(\theta_k)))\cos(\phi(\theta_k)) \leq 0) \end{aligned} \tag{3.18}$$

These constraints are updated at each predicted position of θ and the method has been successfully used for autonomous driving on a track [3].

3.3.2 Slip angle constraint

A constraint on the slip angle is introduced to enhance stability of driving manoeuvres. The constraint is formulated by an approximation of the slip angles in order to make it linear. It is given by:

$$\alpha_{min} \leq \hat{\alpha}_i \leq \alpha_{max}, \; i = f, r \tag{3.19}$$

Where $\hat{\alpha}_f$ is given by:

$$\hat{\alpha}_f = -(\frac{\omega l_f + v_y}{\hat{v}_x^*}) + \delta \tag{3.20}$$

and $\hat{\alpha}_r$ is given by:

$$\hat{\alpha}_r = (\frac{\omega l_r - v_y}{\hat{v}_x^*}) \tag{3.21}$$

Where $\hat{v}_x^*$ is the shifted optimal sequence from the previous solution.

3.3.3 Obstacle avoidance

By modifying the border constraints such that they also geometrically covers the obstacle we can integrate obstacle avoidance into the current setup of the controller. The main problem is how to decide which side of the lane should be adjusted.
In this approach a parameterized polynomial is constructed with one extra degree of freedom. An optimal solution for the polynomial coefficients are found by posing the problem as a constraint-free convex optimisation problem, which there are analytic

solutions for. Then by imposing constraints in the parameter space of the extra degree of freedom, given by the parameter pair $\{a_6, b_6\}$, a sub-optimal solution is found which satisfies the constraints while still guaranteeing kinematic feasibility. Note that notations used in this section do not refer to the same notations as in other sections. The model used is given by:

$$\dot{x} = v\cos\phi \tag{3.22a}$$
$$\dot{y} = v\sin\phi \tag{3.22b}$$
$$\dot{\phi} = \frac{v\tan\delta}{L} \tag{3.22c}$$

Where v is the vehicle speed, δ is the steering angle and L is the length of the car. Adapted from [4] we can define the initial and final configurations for the polynomial:

$$x_0 = x_k^m \tag{3.23a}$$
$$\dot{x}_0 = v_k^m\cos(\phi_k) \tag{3.23b}$$
$$\ddot{x}_0 = a_k^m\cos(\phi_k) - \frac{v_k\tan(\delta_k\sin(\phi_k))}{l} \tag{3.23c}$$
$$x_f = x_{k+N}^p \tag{3.23d}$$
$$\dot{x}_f = v_{k+N}^p\cos(\phi_{k+N}) \tag{3.23e}$$
$$\ddot{x}_f = a_{k+N}^p\cos(\phi_{k+N}) - \frac{v_f\tan(\delta_{k+N}\sin(\phi_{k+N}))}{l} \tag{3.23f}$$
$$y_0 = y_k^m \tag{3.23g}$$
$$\dot{y}_0 = v_k^m\sin(\phi_k) \tag{3.23h}$$
$$\ddot{y}_0 = a_k^m\sin(\phi_k) - \frac{v_k\tan(\delta_k\cos(\phi_k))}{l} \tag{3.23i}$$
$$y_f = y_{k+N}^p \tag{3.23j}$$
$$\dot{y}_f = v_{k+N}^p\sin(\phi_{k+N}) \tag{3.23k}$$
$$\ddot{y}_f = a_{k+N}^p\sin(\phi_{k+N}) - \frac{v_f\tan(\delta_{k+N}\cos(\phi_{k+N}))}{l} \tag{3.23l}$$

Where the superscripts m and p stands for measured and predicted, respectively. The trajectory is then generated by the following polynomials:

$$x(t) = \mathbf{f}(t)(\mathbf{G})^{-1}(\mathbf{E} - \mathbf{H}a_6) + a_6t^6 \tag{3.24a}$$
$$y(t) = \mathbf{f}(t)(\mathbf{G})^{-1}(\mathbf{F} - \mathbf{H}b_6) + b_6t^6 \tag{3.24b}$$

Where

$$\mathbf{f}(t) = [1 \quad t \quad t^2 \quad t^3 \quad t^4 \quad t^5] \tag{3.25a}$$
$$\mathbf{E} = [x_0 \quad \dot{x}_0 \quad \ddot{x}_0 \quad x_f \quad \dot{x}_f \quad \ddot{x}_f] \tag{3.25b}$$
$$\mathbf{F} = [y_0 \quad \dot{y}_0 \quad \ddot{y}_0 \quad y_f \quad \dot{y}_f \quad \ddot{y}_f] \tag{3.25c}$$
$$\mathbf{H} = [t_0^6 \quad 6t_0^5 \quad 30t_0^4 \quad t_f^6 \quad 6t_f^5 \quad 30t_f^4] \tag{3.25d}$$

and

$$\mathbf{G} = \begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & t_f & t_f^2 & t_f^3 & t_f^4 & t_f^5 \\ 0 & 1 & 2t_f & 3t_f^2 & 4t_f^3 & 5t_f^4 \\ 0 & 0 & 2 & 6t_f & 12t_f^2 & 20t_f^3] \end{bmatrix} \tag{3.26}$$

Where t is given by the time over the horizon of the LTVMPCC optimisation problem. A convex optimisation problem is then posed in order to solve for the coefficients. Several different cost functions of interest can be constructed. One can minimize against a straight line connecting the initial and final configurations, thus forming a shortest-path type of polynomial. Another idea would be to minimize the difference between the predicted path given by the controller and the polynomial, thus achieving minimal disruption to the current trajectory of the car. In this thesis we employ a cost function which minimizes the energy consumption. This cost function was chosen since it can be implied that the minimum waste of energy always allows for the most efficient manoeuvre around an obstacle. However no further analysis on the effects of the different cost functions are made. Since $\dot{x}$ and $\dot{y}$ can be represented analytically in terms of a_6 and b_6 we can define a kinematic energy cost function:

$$\min_{a_6, b_6} \quad J_E = \int_{t_0}^{t_f} \frac{1}{2} m(\dot{x}^2 + \dot{y}^2) dt \tag{3.27a}$$

Re-formulating the cost using 3.24 the cost function becomes:

$$J_E = \frac{1}{2} m(s_2(a_6 + \frac{s_1}{2s_2})^2 + s_2(b_6 + \frac{s_4}{2s_2})^2 + s_0 + s_3 - \frac{(s_1^2 + s_4^2)}{4s_2}) \tag{3.28}$$

Where:

$$s_0 = \int_{t_0}^{t_f} (\dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{E})^2 dt \tag{3.29a}$$

$$s_1 = 2\int_{t_0}^{t_f} (6t^5 - \dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{H})(\dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{E}) dt \tag{3.29b}$$

$$s_2 = \int_{t_0}^{t_f} (6t^5 - \dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{H})^2 dt \tag{3.29c}$$

$$s_3 = \int_{t_0}^{t_f} (\dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{F})^2 dt \tag{3.29d}$$

$$s_4 = \int_{t_0}^{t_f} (6t^5 - \dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{H})(\dot{\mathbf{f}}(t)\mathbf{G}^{-1}\mathbf{F}) dt \tag{3.29e}$$

And where $\dot{\mathbf{f}}(t) = [0 \quad 1 \quad 2t \quad 3t^2 \quad 4t^3 \quad 5t^4]$. Examining 3.28 it is clear that the optimal minimizing solution lies at:

$$a_6^* = -\frac{s_1}{2s_2} \tag{3.30a}$$

$$b_6^* = -\frac{s_4}{2s_2} \tag{3.30b}$$

In order for the trajectory to avoid obstacles as well as stay within the track two constraints are introduced. The obstacle constraint is formulated as an inequality which prohibits the trajectory to be inside a prohibited circular area:

$$(x(t) - x_i)^2 + (y(t) - y_i)^2 \geq (r + r_i)^2 \tag{3.31}$$

The subscript i denotes the i:th obstacle radius and r is set to half the width of the car. The obstacles are considered static in the above formulation, however the terms x_i and y_i can be reformulated to include relative velocity between the car and the obstacle in order to consider the a moving obstacle. Even though the obstacles are formally considered static, they can in fact be non-static as the polynomial trajectory is re-calculated at each time sample. Using 3.24 the inequality can be formulated as an inequality in the parameter space:

$$(a_6 + \frac{g_{1,i}(t)}{g_{2,i}(t)})^2 + (b_6 + \frac{g_{3,i}(t)}{g_{2,i}(t)})^2 \geq \frac{(r + r_i)^2}{g_{2,i}(t)^2} \tag{3.32}$$

Where

$$g_1(t) = \mathbf{f}(t)\mathbf{G}^{-1}\mathbf{E} - x_i \tag{3.33a}$$
$$g_2(t) = t^6 - \mathbf{f}(t)\mathbf{G}^{-1}\mathbf{H} \tag{3.33b}$$
$$g_3(t) = \mathbf{f}(t)\mathbf{G}^{-1}\mathbf{F} - y_i \tag{3.33c}$$

In addition to avoiding any obstacles, the trajectory must also stay within the lane. We introduce such a constraint as:

$$(x(t) - x_d(\hat{\Theta}_k^*))^2 + (y(t) - y_d(\hat{\Theta}_k^*))^2 \leq (r_{width})^2 \tag{3.34}$$

The inequality ensures that at each time sample the trajectory stays within a circle centred at $\{x_d, y_d\}$ given by the predicted $\hat{\Theta}^*$ trajectory. And r_{width} is set to half of the width of the lane. The inequality is then reformulated into the parameter space in the same way as the obstacle constraint which gives the inequality:

$$(a_6 + \frac{l_{1,i}(t)}{g_{2,i}(t)})^2 + (b_6 + \frac{l_{3,i}(t)}{g_{2,i}(t)})^2 \leq \frac{(r_{width})^2}{g_{2,i}(t)^2} \tag{3.35}$$

Where

$$l_1(t) = \mathbf{f}(t)\mathbf{G}^{-1}\mathbf{E} - x_d(\hat{\Theta}_k^*) \tag{3.36a}$$
$$l_3(t) = \mathbf{f}(t)\mathbf{G}^{-1}\mathbf{F} - y_d(\hat{\Theta}_k^*) \tag{3.36b}$$

The set of trajectories which avoids the given obstacles and stays within the lane can now be defined by finding the solution for $\{a_6, b_6\}$ in the feasible area of the parameter space. See Figure 3.3 and Figure 3.4.

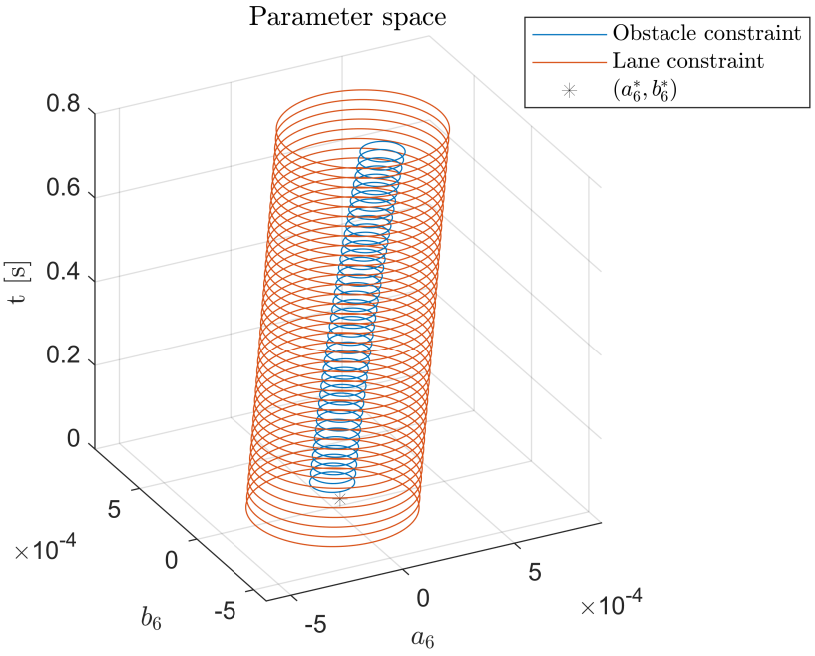


Figure 3.3: Parameter space - ISO view

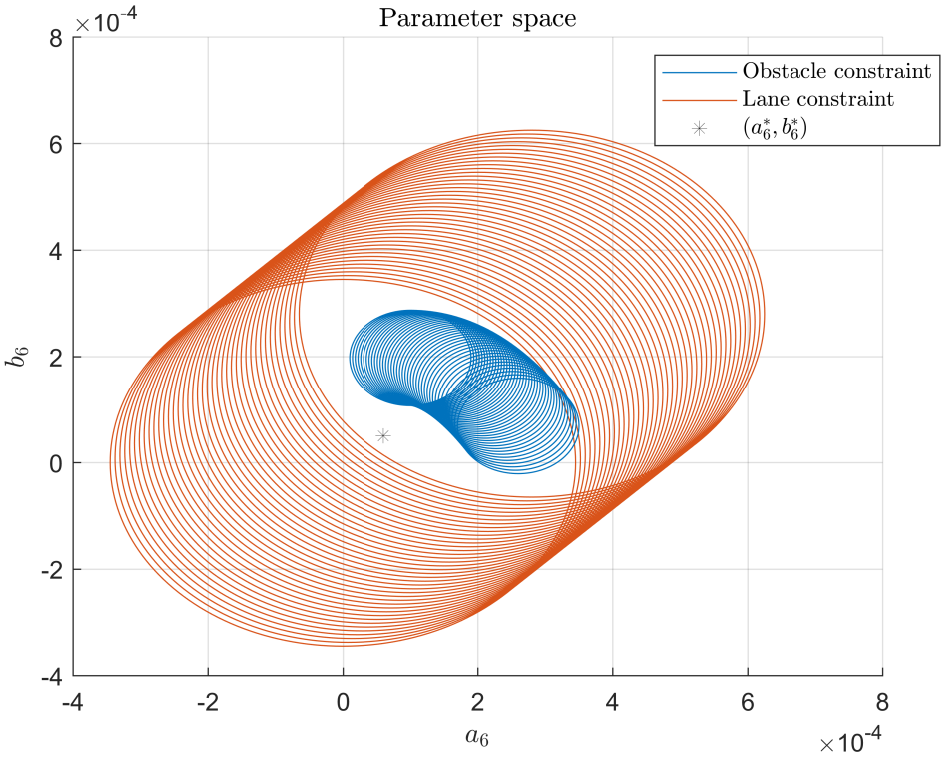


Figure 3.4: Parameter space - Top view

Finding such a solution requires to solve a constrained non-linear optimisation problem, unless of course the optimal solution $\{a_6^*, b_6^*\}$ already lies in this space as can be seen in figure 3.4. In practice this means that the optimal trajectory generated lies inside the lane and is not obstructed by an obstacle. However if this is not the case then a sub-optimal solution must be found. Such a solution can be found analytically by fixing the solution to a straight line, i.e. at each time t fix one parameter and solve for the other. In figure 3.5 b_6 has been fixed to $b_6 = b_6^*$ while the set of feasible solutions for a_6 at each time step is denoted A_6 and is marked green.

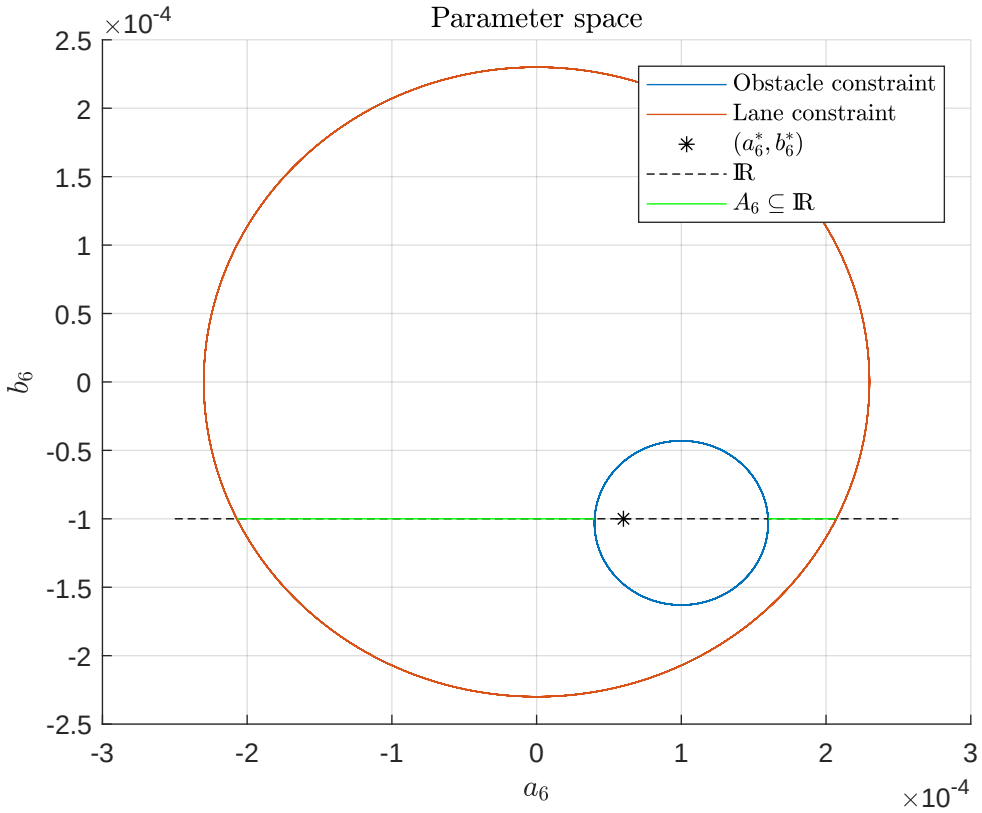


Figure 3.5: Sub-optimal solution range.

When the sub-optimal solution has been found for a trajectory which stays inside the lane and avoids the given obstacle we can compute the lane constraint adjustments. These are computed by first determining which side the trajectory passes the i:th obstacle. This is done by rotating the points of the trajectory and current obstacle to $\phi = 0$ and compare their y-coordinates. Then it is determined which points in the $\hat{\Theta}^*$ sequence lies inside the obstacle circle projected onto the centre line. Finally at each of these points the border adjustment magnitude is obtained by computing the intersection between a straight line the current obstacle circle. See Figure 3.6 for an example of a computed polynomial and the corresponding border adjustment.

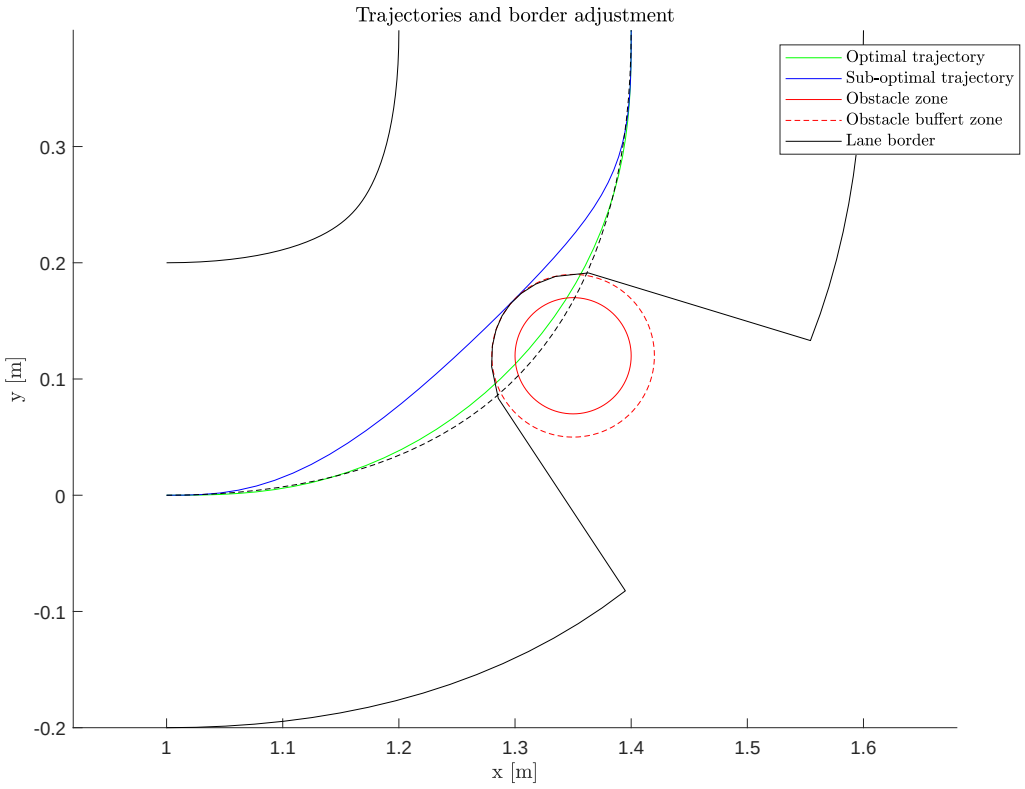


Figure 3.6: Border adjustment using polynomial path planner.

3.3.4 Contraction constraint

In order to stabilize the closed loop an additional state constraint is introduced, a contraction constraint. This is a Lyapunov based approach to stability in that a Lyapynov function is chosen and then constrained to decrease in discrete time. The Lyapynov function is the constraint itself, which is chosen as the weighted norm of the state vector. The constraint is given by:

$$||(\boldsymbol{\xi}^{j+1} - \boldsymbol{\xi}_{sp})||_{\mathbf{P}} < \alpha||(\boldsymbol{\xi}^{j} - \boldsymbol{\xi}_{sp})||_{\mathbf{P}} \tag{3.37}$$

Where the norm applied is a so called weighted norm: $||\boldsymbol{\xi}||_P = \sqrt{\boldsymbol{\xi}^T \mathbf{P} \boldsymbol{\xi}}$. The vector $\boldsymbol{\xi}_{sp}$ is any equilibrium point where one wishes the system to contract towards, α is the contraction parameter, $\alpha \in [0,1)$ and $\mathbf{P} \succ 0$. Note that the superscript on the state vector $\boldsymbol{\xi}$ is the so called contraction step notation. It is introduced to clarify that the constraint is held constant for a period of N time steps. The norm is allowed to increase in between, but is constrained to decrease with respect to sections of N time steps. Figure 3.7 shows the norm of the plant and the norm of the prediction model during a horizon (dashed lines) at chosen time steps.

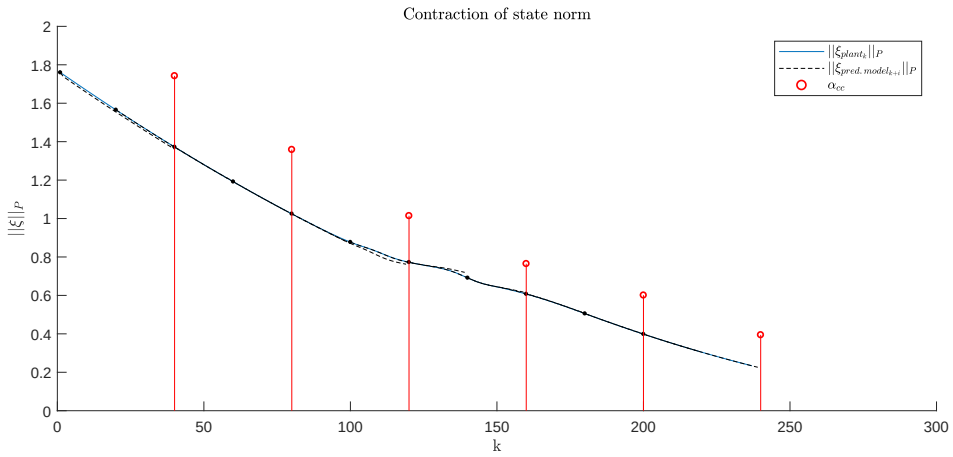


Figure 3.7: Contraction constraint with $N = 40$.

where j is the contraction step, k is the time step and i is the prediction step. Figure 3.7 shows how the constraint is held constant over a section of N time steps, i.e the constraint is fixed in time and the number of time steps between the current time step and the position of the contraction constraint on the time scale decreases. When $k = N-1$, the constraint is updated, along with the contraction step $j = j+1$ and then placed at jN on the time scale.
As can be seen as long as the initial optimisation problem at time k is feasible, then the consecutive $N-1$ optimisation problems are feasible as well with respect to the constraint, disregarding disturbances. The contraction parameter α can in practice be seen as time varying in case large decaying disturbances enter the system between j and $j+1$, such that infeasibilities in that scenario can be avoided by increasing α. In order to account for the system being a linear approximation of the actual nonlinear system, more restrictive bounds on the contraction parameter can be analytically found, that is: $0 \leq \underline{\alpha} \leq \alpha \leq \bar{\alpha} < 1$. The upper bound $\bar{\alpha}$ accounts for the model/plant mismatch due to linearization and is thus in practice computed at every contraction step. Following [2] these bounds on α are given by, where the discrete time subscripts have been dropped for increased readability:

$$\bar{\alpha} = 1 - \frac{\gamma e^{LNT_s}(e^{LNT_s} - 1))}{L} \tag{3.38}$$

Where L is a Lipschitz discrete time varying constant for our system and γ is a discrete time varying constant which quantifies the "strength" of the nonlinearities in our system. The Lipschitz condition for dynamical systems is given by:

$$||\mathbf{f}(\boldsymbol{x}_1) - \mathbf{f}(\boldsymbol{x}_2)|| \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2|| \tag{3.39}$$

Where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ is any state vector in the bounded set $\mathcal{X}$. For our linear time varying system the left hand side of 3.39 can be re-written as:

$$\begin{aligned} ||\mathbf{f}(\boldsymbol{x}_1) - \mathbf{f}(\boldsymbol{x}_2)|| =& ||\mathbf{A}_c\boldsymbol{x}_1 + \mathbf{B}_c\mathbf{u} - (\mathbf{A}_c\boldsymbol{x}_2 + \mathbf{B}_c\mathbf{u}) = \\ =& ||\mathbf{A}_c\boldsymbol{x}_1 + \mathbf{B}_c\mathbf{u} - \mathbf{A}_c\boldsymbol{x}_2 - \mathbf{B}_c\mathbf{u}|| = \\ =& ||\mathbf{A}_c\boldsymbol{x}_1 - \mathbf{A}_c\boldsymbol{x}_2|| = ||\mathbf{A}_c|| \cdot ||\boldsymbol{x}_1 - \boldsymbol{x}_2|| \end{aligned}$$

If we plug this into the standard Lipschitz condition 3.39 then we get:

$$||\mathbf{A}_c|| \cdot ||\boldsymbol{x}_1 - \boldsymbol{x}_2|| \leq L||\boldsymbol{x}_1 - \boldsymbol{x}_2||$$

And it is clear to see that the Lipschitz constant for a linear time varying system is only depending on the norm of the linearized $\mathbf{A}$ matrix, i.e:

$$||\mathbf{A}_c|| \leq L \tag{3.40}$$

Further assumptions are imposed from [2]. For all $\boldsymbol{x} \in \mathcal{X}_k$, all $\mathbf{u} \in \mathcal{U}$ the following bounds must hold over the prediction horizon:

Assumption 1:

$$||\mathbf{g}_c + \mathbf{A}_c\boldsymbol{x} + \mathbf{B}_c\mathbf{u}||_{\mathbf{P}} \leq L(||\boldsymbol{x}||_{\mathbf{P}} + ||\mathbf{u}||) \tag{3.41}$$

Where $\mathbf{A}_c, \mathbf{B}_c$ and $\mathbf{g}_c$ are the linearized matrices for the system in continuous time. The inequality is instead computed with the predicted state trajectory over the horizon along with its corresponding optimal control input.

Assumption 2:

$$||\mathbf{f}(\boldsymbol{x}_k^p, \mathbf{u}_k) - \mathbf{f}(\boldsymbol{x}_k, \mathbf{u}_k)||_{\mathbf{P}} \leq L(||\boldsymbol{x}_k^p - \boldsymbol{x}_k||_{\mathbf{P}}) \tag{3.42}$$

Where $\boldsymbol{x}_k^p$ is the plant state at time k and $\mathbf{f}$ is the nonlinear system. The predicted state is used for computation. The term $\mathbf{f}(\boldsymbol{x}_k, \mathbf{u}_k)$ can be re-written as $\mathbf{f}(\boldsymbol{x}_k, \mathbf{u}_k) = \mathbf{A}_c\boldsymbol{x}_k + \mathbf{B}_c\mathbf{u}_k + \mathbf{g}_c + \mathbf{F}(\boldsymbol{x}_k, \mathbf{u}_k)$ where $\mathbf{F}$ is the higher order terms of the Taylor expansion of $\mathbf{f}$.

The idea is to find a Lipschitz constant for the transient states between the contraction steps, i.e between $j \to j+1$. In this thesis we find the Lipschitz constant by construction, that is we compute all Lipschitz constants over the prediction horizon from 3.40 then remove those who does not satisfy 3.41 and 3.42.

Assumption 3:

$$||\mathbf{F}(\boldsymbol{x}, \mathbf{u})||_{\mathbf{P}} \leq \gamma(||\boldsymbol{x}||_{\mathbf{P}} + ||\mathbf{u}||) \tag{3.43}$$

The inequality is computed with the predicted state trajectory over the horizon along with its corresponding optimal control input. The parameter $\gamma \in [0, \infty)$ is a positive constant which quantifies the norm increase in the higher order terms of the Taylor expansion of $\mathbf{f}$ given the state and applied input. This can been seen as quantifying the magnitude of the nonlinearities in the system. In this thesis we restrict ourselves to only the quadratic higher order terms, the rest of the higher order terms are neglected as they become insignificantly small. The constant γ is computed by construction, that is by dividing the right hand side of 3.43, excluding γ, with the left hand side. This is repeated for all prediction steps $k+1 \to k+N$ and then the largest value of those computed γ are chosen. The largest value is chosen since we want to find a bound for the transient states between the contraction steps

and this should be done in a worst case scenario manner.

Since $\alpha \geq 0$ then it is clear that there exists a condition that must be satisfied in order for the computed bound to be valid. Since:

$$\bar{\alpha} = 1 - \frac{\gamma e^{LNT_s}(e^{LNT_s} - 1))}{L}$$

Then in order for $\alpha \geq 0$ to be satisfied then:

$$\frac{\gamma e^{LNT_s}(e^{LNT_s} - 1))}{L} \leq 1$$

Rewriting this gives a condition w.r.t γ

Condition of existence:

$$\gamma < \frac{L}{e^{LNT_s}(e^{LNT_s} - 1)} \tag{3.44}$$

After having computed all Lipschitz constants over the prediction horizon, excluded those who do not satisfy the assumptions we are left with a range of constants which needs to satisfy the condition of existence 3.44. Figure 3.8 shows the plot of the right hand side of 3.44 and the upper bound $\bar{\alpha}$ for a value $\gamma = 0.009$. It can be seen that where the lines cross there no longer exists a Lipschitz constant L which will yield a feasible $\bar{\alpha}$. It can also be seen from Figure 3.8 that the smallest L which satisfy condition 3.44 will also yield the least restrictive upper bound $\bar{\alpha}$. Hence, in this work, we take $min(\boldsymbol{L})$ of the vector of Lipschitz constants which satisfy 3.41, 3.42 and finally 3.44. This way of applying the conditions by the use of predicted trajectories is different than in [2] where the author suggests searching the whole state space. This is not feasible for implementation with the aim for real time embedded application. Thus we can not guarantee that the conditions hold for all $\boldsymbol{x} \in \mathcal{X}$ and all $\boldsymbol{u} \in \mathcal{U}$. However the context of the contraction constraint is to bound the transient states between $j \to j+1$, that is over the coming N time steps. Thus the predicted trajectories at time k will yield a good indication of where the system will be in the state space for the coming time window where the transient states must be bounded and contracted enough to impose stability.

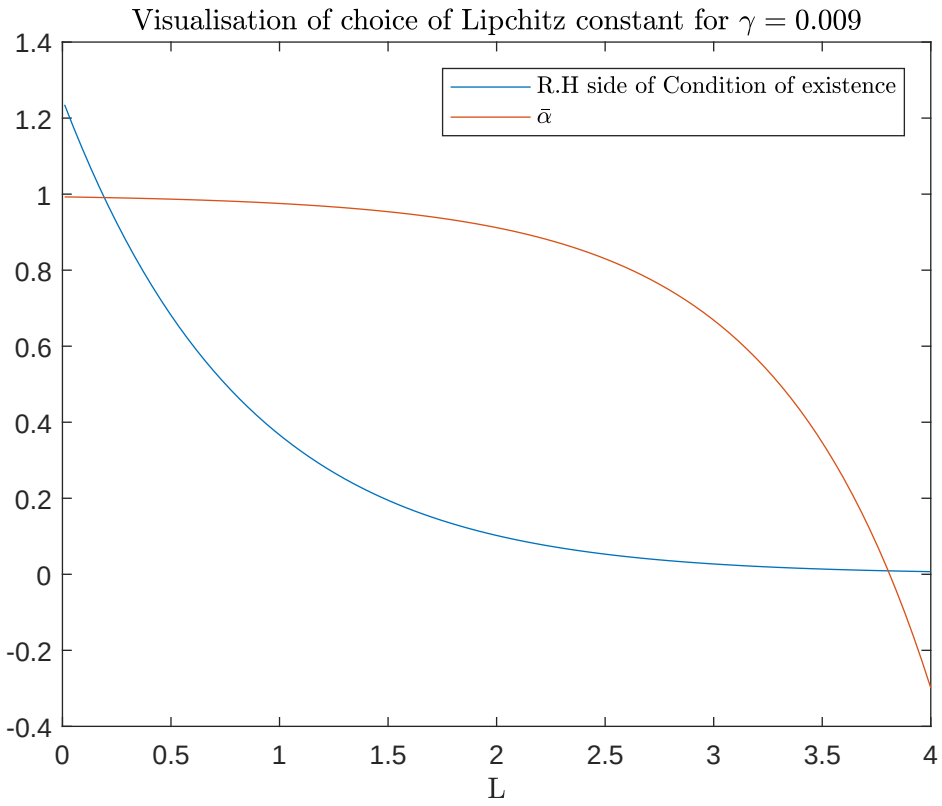


Figure 3.8: Function values over a typical range of Lipschitz constant L.

A lower bound $\underline{\alpha}$ can be computed in order to prevent the contraction constraint of driving the system into an infeasible set. From [2] it can be seen that computing this lower bound can only be done under a strict assumption that $\mathbf{A}_c = \frac{\partial \mathbf{f}}{\partial \boldsymbol{x}}\Big|_{(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*)}$ is stable, i.e all $Re(\lambda)$ are located in the left half plane, for all $(\hat{\boldsymbol{x}}^*, \hat{\mathbf{u}}^*)$ which the linearization is performed around. The introduction of θ in to the system dynamics yields a positive eigenvalue in the matrix A_c and thus prevents us from computing this lower bound on α. In order to prevent the upper bound $\bar{\alpha}$ to drive the system into infeasibility we simply set a constant lower bound to $\underline{\alpha} = 0.7$.
This stability approach was chosen as it has minimal effect on the controller. Firstly, the contraction is only enforced a intervals of N time steps, hence it is not very difficult to fulfil. Secondly, since $\theta \in [-L_{track}, 0]$ and the norm is weighted we can achieve a naturally decreasing norm of the state vector as long as we move along the path. Thus, unless stated otherwise, P is given by:

$$\mathbf{P} = \begin{bmatrix} 10^{-5} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10^{-5} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 10^{-5} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 10^{-5} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 10^{-5} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 10^{-5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.45}$$

3.4 Casting into a QP problem

The optimisation problem 3.13b can be casted into a Quadratically Constrained Quadratic Program (QCQP) and be solved with readily available solvers. For this work the optimisation suite CPLEX have been used to solve the problem, both in simulation and implementation as it has interface to both MATLAB and other programming languages such as C or $C++$.

$$\min_{\mathbf{z}} \quad \mathbf{z}^T\mathbf{H}\mathbf{z} + \mathbf{f}^T\mathbf{z} \tag{3.46a}$$

$$\text{s.t.} \quad \mathbf{A}_{eq}\mathbf{z} = \mathbf{b}_{eq} \tag{3.46b}$$

$$\mathbf{A}_{in}\mathbf{z} \leq \mathbf{b}_{in} + \boldsymbol{\epsilon} \tag{3.46c}$$

$$\mathbf{z}^T\mathbf{Q}_{qc}\mathbf{z} + \mathbf{l}_{qc}\mathbf{z} \leq r_{qc} \tag{3.46d}$$

Where the decision vector $\mathbf{z}$ is given by:

$$\mathbf{z} = [\boldsymbol{\xi}_{k+1}^T \dots \boldsymbol{\xi}_{k+N}^T \quad \bar{\mathbf{u}}_k^T \dots \bar{\mathbf{u}}_{k+N-1}^T \quad \boldsymbol{\epsilon}_{k+1}^T \dots \boldsymbol{\epsilon}_{k+N}^T]^T \tag{3.47}$$

As can be seen, the problem 3.46 is formulated as a sparse Quadratic Program. Computational cost grows linearly with N in a sparse formulation compared to quadratically or even cubically (depending on which method the chosen solver uses) in a dense formulation [5]. As this work will use a rather large horizon N the sparse formulation is best suited.

The structure of the Hessian is given by:

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}^{\xi}_{k+1|k} & \mathbf{0} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & & & & & & & & \vdots \\ \vdots & \ddots & \mathbf{H}^{\xi}_{k+N|k} & \mathbf{0} & & & & & & & \vdots \\ \vdots & & \mathbf{0} & \mathbf{R}_{\Delta} & -\mathbf{R}_{\Delta} & & & & & & \vdots \\ \vdots & & & -\mathbf{R}_{\Delta} & 2\mathbf{R}_{\Delta} & \ddots & & & & & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & & & & \vdots \\ \vdots & & & & & \ddots & 2\mathbf{R}_{\Delta} & -\mathbf{R}_{\Delta} & & & \vdots \\ \vdots & & & & & & -\mathbf{R}_{\Delta} & \mathbf{R}_{\Delta} & \mathbf{0} & & \vdots \\ \vdots & & & & & & & \mathbf{0} & I_{q_{\epsilon 2}} & \ddots & \vdots \\ \vdots & & & & & & & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \mathbf{0} & I_{q_{\epsilon 2}} \end{bmatrix} \tag{3.48}$$

Due to the need to linearize the contour error e_c and lag error e_l the part of the Hessian which denotes the entries for the state variables H^{ξ} is time varying, and is given by:

$$\mathbf{H}^{\xi}_{k+i|k} = \begin{bmatrix} \circ_1 & \circ_2 & 0 & 0 & 0 & 0 & \circ_3 \\ \circ_4 & \circ_5 & 0 & 0 & 0 & 0 & \circ_6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \circ_7 & \circ_8 & 0 & 0 & 0 & 0 & \circ_9 \end{bmatrix} \tag{3.49}$$

The entries $\circ_i, \quad i = 1, 2..9$ are given in Appendix A.

The diagonal elements of the Hessian was then perturbed by a factor of 10^{-8} to ensure positive definiteness. The linear part of the QP problem is given by the vector $\mathbf{f}$:

$$\mathbf{f} = [\mathbf{f}^{\xi}_{k+1|k} \quad \mathbf{f}^{\xi}_{k+2|k} \cdots \mathbf{f}^{\xi}_{k+N|k} \quad 0 \cdots 0 \quad q_{\epsilon} \cdots q_{\epsilon}]^T \tag{3.50}$$

Where just as the Hessian, $\mathbf{f}$ is time varying to due to the linearization of the contour error e_c and lag error e_l. In particular, the time varying part is f_{ξ} which denotes the entries for the state variables and is given by:

$$\mathbf{f}^{\xi}_{k+i|k} = [\diamond_1 \ \diamond_2 \ 0\,0\,0\,0 \diamond_3]^T \tag{3.51}$$

The individual entries $\diamond_i, \quad i = 1, 2, 3$ can be found in Appendix B.
The equality constraint is constructed by the time varying matrix $\mathbf{A}_{eq}$:

$$\mathbf{A}_{eq} = \begin{bmatrix} \mathbf{A}_{eq1} & \mathbf{A}_{eq2} \end{bmatrix} \tag{3.52}$$

$$\mathbf{A}_{eq1} = \begin{bmatrix} \mathbf{I}_7 & \mathbf{0} & \cdots & \cdots & \cdots & \cdots & 0 \\ -\bar{\mathbf{A}}_{k+1|k} & \mathbf{I}_7 & \mathbf{0} & & & & \vdots \\ \mathbf{0} & -\bar{\mathbf{A}}_{k+2|k} & \ddots & \ddots & & & \vdots \\ \vdots & \mathbf{0} & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \mathbf{0} & \vdots \\ \vdots & & & \ddots & \ddots & \mathbf{I}_7 & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} & -\bar{\mathbf{A}}_{k+N|k} & \mathbf{I}_7 \end{bmatrix} \tag{3.53}$$

$$\mathbf{A}_{eq2} = \begin{bmatrix} -\bar{\mathbf{B}}_{k|k} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & -\bar{\mathbf{B}}_{k+1|k} & \mathbf{0} & & & \vdots \\ \mathbf{0} & \mathbf{0} & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} & -\bar{\mathbf{B}}_{k+N|k} \end{bmatrix} \tag{3.54}$$

The right hand side of the equality constraint is given by the time varying vector $\mathbf{b}_{eq}$:

$$\mathbf{b}_{eq} = \begin{bmatrix} \bar{\mathbf{A}}_{k|k}\,\xi_0 + \bar{\mathbf{g}}_{k|k} \\ \bar{\mathbf{g}}_{k+1|k} \\ \bar{\mathbf{g}}_{k+2|k} \\ \vdots \\ \vdots \\ \bar{\mathbf{g}}_{k+N|k} \end{bmatrix} \tag{3.55}$$

The left hand side of the inequality constraint is given by:

$$\mathbf{A}_{in} = \begin{bmatrix} \mathbf{A}_{in_1} & \mathbf{A}_{in2} & -\mathbf{I}_{6N} \end{bmatrix} \tag{3.56}$$

$$\mathbf{A}_{in1} = \begin{bmatrix} \mathbf{A}^{\xi}_{in_{k+1|k}} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^{\xi}_{in_{k+2|k}} & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} & \mathbf{A}^{\xi}_{in_{k+N|k}} \end{bmatrix} \tag{3.57}$$

$$\mathbf{A}_{in2} = \begin{bmatrix} \mathbf{A}_{in}^{\bar{u}} & \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} \\ \mathbf{0} & \ddots & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \cdots & \cdots & \mathbf{0} & \mathbf{A}_{in}^{\bar{u}} \end{bmatrix} \tag{3.58}$$

Where the time varying matrix $\mathbf{A}_{in}^{\xi}$ is given by:

$$\mathbf{A}_{in}^{\xi} = \begin{bmatrix} \sin\phi(\hat{\theta}^*_{k+i|k}) & -\cos\phi(\hat{\theta}^*_{k+i|k}) & 0 & 0 & 0 & 0 & 0 \\ -\sin\phi(\hat{\theta}^*_{k+i|k}) & \cos\phi(\hat{\theta}^*_{k+i|k}) & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\hat{v}_x^*} & -\frac{l_f}{\hat{v}_x^*} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\hat{v}_x^*} & \frac{l_f}{\hat{v}_x^*} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\hat{v}_x^*} & \frac{l_r}{\hat{v}_x^*} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\hat{v}_x^*} & -\frac{l_r}{\hat{v}_x^*} & 0 \end{bmatrix} \tag{3.59}$$

The part of the $\mathbf{A}_{in}$ matrix which corresponds to the input variables is given by:

$$\mathbf{A}_{in}^{\bar{u}} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{3.60}$$

The time varying right hand side of the inequality constraint is given by:

$$\mathbf{b}_{in} = \begin{bmatrix} x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}) - y_d(\hat{\theta}^*_{k+i|k})\cos\phi(\hat{\theta}^*_{k+i|k}) + \frac{w}{2} - d_{right} \\ -x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}) + y_d(\hat{\theta}^*_{k+i|k})\cos\phi(\hat{\theta}^*_{k+i|k}) + \frac{w}{2} - d_{left} \\ \alpha_{max} \\ \alpha_{max} \\ \alpha_{max} \\ \alpha_{max} \end{bmatrix} \tag{3.61}$$

Where d_{right} and d_{left} stands for the distance of border adjustments, right and left, respectively. The distances comes as input to the controller function from the obstacle avoidance path planner. α_{max} appears in all four entries for the slip angle constraint due to the reformulation of the symmetrical box constraint 3.19.
The stability constraint is formulated as a quadratic constraint and is incorporated using the following matrices:

$$\mathbf{Q}_{qc} = \begin{bmatrix} 0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & \ddots & & & & & \vdots \\ \vdots & & \ddots & & & & \vdots \\ \vdots & & & \ddots & & & \vdots \\ \vdots & & & & \ddots & & \vdots \\ \vdots & & & & & 0 & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & \mathbf{P} \end{bmatrix} \tag{3.62}$$

Where the position of the matrix $\mathbf{P}$ along the diagonal depends on the time step k and contractive step j, as described in section 3.3.4.

$$\mathbf{l}_{qc} = [0 \cdots 0 \quad -2P\boldsymbol{\xi}_{sp}]^T \tag{3.63}$$

$$r_{qc} = \bar{\alpha}||\boldsymbol{\xi} - \boldsymbol{\xi}_{sp}||_{\mathbf{P}} \tag{3.64}$$

Where the position of the entries in the $\mathbf{l}_{qc}$ corresponds to the position of $\mathbf{P}$ along the diagonal in the $\mathbf{Q}_{qc}$ matrix.

3.5 Baseline controller

For comparison purposes a base controller was designed. The main controller in this thesis utilises a cost function which is difficult to compare with a more simplistic controller. The latter controller will also use an offline calculated reference trajectory.

3.5.1 Reference trajectory

Since this work is focused on autonomous high speed manoeuvres a natural choice for the reference trajectory is a time optimal one. This trajectory is generated by solving the following continuous time optimisation problem:

$$\begin{aligned} \min \quad & \int_{t_0}^{t_f} dt && \text{(3.65a)} \\ \text{s.t.} \quad & \dot{\boldsymbol{x}} = \mathbf{f}(\boldsymbol{x}(t), \mathbf{u}(t)) && \text{(3.65b)} \\ & \boldsymbol{x}(t) \in \mathcal{X} && \text{(3.65c)} \\ & \mathbf{u}(t) \in \mathcal{U} && \text{(3.65d)} \end{aligned}$$

Equation 3.65 states that the final time shall be minimised subject to the full six state nonlinear system dynamics given by 2.1, state constraints as well as control input constraints. Given the complexity of the nonlinear system dynamics and the use of time varying state constraints it is difficult so solve analytically. Instead, Equation 3.65 is re-casted into an equivalent discrete time optimisation problem. This type of optimisation problem is solved over a finite, and fixed, number of time

steps. Hence to minimise time we apply a re-formulation of time. Let $\sigma \in [0, 1]$ denote the new normalized time variable and let $\sigma = \frac{t}{t_f}$. This yields $\frac{d}{d\sigma} = t_f \frac{d}{dt}$. A new state can now be introduced, $r = t_f$ and let $\dot{r} = 0$. The system dynamics given by 2.1 is now re-written and augmented by the new state r:

$$\dot{x}r = v_x \cos\varphi - v_y \sin\varphi \tag{3.66a}$$
$$\dot{y}r = v_x \sin\varphi + v_y \cos\varphi \tag{3.66b}$$
$$\dot{\varphi}r = \omega \tag{3.66c}$$
$$\dot{v}_x r = \frac{1}{m}(F_{r,x} + F_{f,x}\cos\delta - F_{f,y}\sin\delta + m v_y w) \tag{3.66d}$$
$$\dot{v}_y r = \frac{1}{m}(F_{r,y} + F_{r,x}\sin\delta + F_{f,y}\cos\delta - m v_x w) \tag{3.66e}$$
$$\dot{\omega}r = \frac{1}{I_z}(l_f F_{f,y}\cos\delta - l_r F_{r,y}) \tag{3.66f}$$
$$\dot{r} = 0 \tag{3.66g}$$

Giving the new state vector $\bar{\boldsymbol{x}} = [x \ y \ \varphi \ v_x \ v_y \ \omega \ r]^T$. The problem can now be formulated as a discrete time NLP, given by:

$$\min \quad \sum_{k=1}^{P} r \tag{3.67a}$$
$$\text{s.t.} \quad \bar{\boldsymbol{x}}_{k+1} = \mathbf{f}(\bar{\boldsymbol{x}}_k, \mathbf{u}_k) \tag{3.67b}$$
$$\bar{\boldsymbol{x}}_k \in \mathcal{X}_k \tag{3.67c}$$
$$\mathbf{u}_k \in \mathcal{U} \tag{3.67d}$$

Where r is the newly added seventh state variable corresponding to t_f and P is the total number of discretization points. The system dynamics are integrated and re-scaled back using the Euler method with $T_s = \frac{\frac{t_f}{P}}{t_f} = \frac{1}{P}$. Additional state constraints are imposed. A circular constraint keeps the trajectory inside the track:

$$(x_k - x_\theta)^2 + (y_k - y_\theta)^2 \leq (r_{track} - \frac{w_{car}}{2})^2 \tag{3.68}$$

Where x_θ and y_θ are the closest pair of points on the centre line to the car's position, r_{track} is the width of half the track and w_{car} is the width of the car. Another circular constraint is added for obstacle avoidance:

$$-((x_k - x_\theta)^2 + (y_k - y_\theta)^2 \leq (r_{obstacle} + \frac{w_{car}}{2})^2) \tag{3.69}$$

Where $r_{obstacle}$ is the radius of the prohibited zone. Half the width of the car is added since the constraint is formulated with respect to the centre point of the car. Slip angle constraints are added similar to 3.19 with the same upper and lower bounds. The difference in this formulation is that there is no need to approximate the slip angles equations to make them linear. Initial and final conditions are imposed on x, y and φ given by θ_0 and θ_P, respectively. Finally, the same control input constraints

as in 3.14 are imposed. The time optimal problem 3.67 is then solved using the $MATLAB$ function $fmincon$ with the trajectory given by the centre line as initial guess.

3.5.2 Time varying LQ controller

Since 3.67 generates reference trajectories for all states and inputs it is sufficient to use a simple discrete time Linear Quadratic controller as a baseline controller. The LQ controller minimises the following quadratic cost function:

$$\min \quad \sum_{k=1}^{\infty} \mathbf{x}_k^T \mathbf{Q}_{lq} \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R}_{lq} \mathbf{u}_k \tag{3.70a}$$

For a linear time invariant system. However since the system used here is nonlinear it has to be linearized at every time step k. The linear system is given by:

$$\begin{aligned} \dot{\boldsymbol{x}} \approx \mathbf{f}(\mathbf{x}^*, \mathbf{u}^*) + \nabla \mathbf{f}(\mathbf{x}^*, \mathbf{u}^*) \begin{bmatrix} \mathbf{x} - \mathbf{x}^* \\ \mathbf{u} - \mathbf{u}^* \end{bmatrix} = \mathbf{f}(\mathbf{x}^*, \mathbf{u}^*) + \\ \mathbf{A}_c(\mathbf{x} - \mathbf{x}^*) + \mathbf{B}_c(\mathbf{u} - \mathbf{u}^*) = \mathbf{A}_c \tilde{\mathbf{x}} + \mathbf{B}_c \tilde{\mathbf{u}} \end{aligned} \tag{3.71}$$

Where $\mathbf{x}^*$ and $\mathbf{u}^*$ are the optimal reference trajectories. In contrast to the linearization used for the Model Predictive Contouring Controller earlier this uses the deviation variables to avoid producing an affine system term. This facilitates the computation of the LQ gain $\mathbf{K}_{lq}$. The gain is given by:

$$\mathbf{K}_{lq} = (\mathbf{B}_d^T \mathbf{P}_{lq} \mathbf{B}_d + \mathbf{R}_{lq})^{-1} (\mathbf{B}_d^T \mathbf{P}_{lq} \mathbf{A}_d) \tag{3.72}$$

Where $\mathbf{P}_{lq}$ is the solution to the discrete time Riccati equation. The optimal control is then given by:

$$\tilde{\mathbf{u}}^o = -\mathbf{K}_{lq} \tilde{\boldsymbol{x}} \tag{3.73}$$

The system equations are discretized in the same manner as the LTVMPCC and the $MATLAB$ function $dlqr$ is then used to compute the optimal gain $\mathbf{K}_{lq}$ at each time step k. Since deviation variables are used for computation of the gain, the optimal control must be re-formulated into absolute variables:

$$\tilde{\mathbf{u}}^o = -\mathbf{K}_{lq} \tilde{\mathbf{x}} \Rightarrow \mathbf{u}^o - \mathbf{u}^* = -\mathbf{K}_{lq}(\mathbf{x} - \mathbf{x}^*) \Rightarrow \mathbf{u}^o = \mathbf{u}^* - \mathbf{K}_{lq}(\mathbf{x} - \mathbf{x}^*) \tag{3.74}$$

3.5.2.1 $\mathbf{Q}_{lq}$ and $\mathbf{R}_{lq}$ weights

An important aspect of how well the LQ controller will track the reference trajectories are the weighting matrices $\boldsymbol{Q}_{lq}$ and $\boldsymbol{R}_{lq}$. These will tune the behaviour of the controller. There are some rules of thumb to go by when tuning these. The widely known Bryson's rule is often used. It states that the diagonal entries in $\boldsymbol{Q}_{lq}$ and $\boldsymbol{R}_{lq}$ should be the reciprocals of the square of the maximum allowed value for the state and control input. That is though just the start of a trial and error process. In this work we instead employ a stochastic optimisation algorithm to find the weights. The

algorithm used is Particle Swarm Optimisation, or PSO. It is an algorithm which mimics the behaviour of an animal swarm where each individual moves with relative unity with respect to the swarm. The flow diagram shown in figure 3.9 visualises how the algorithm works:

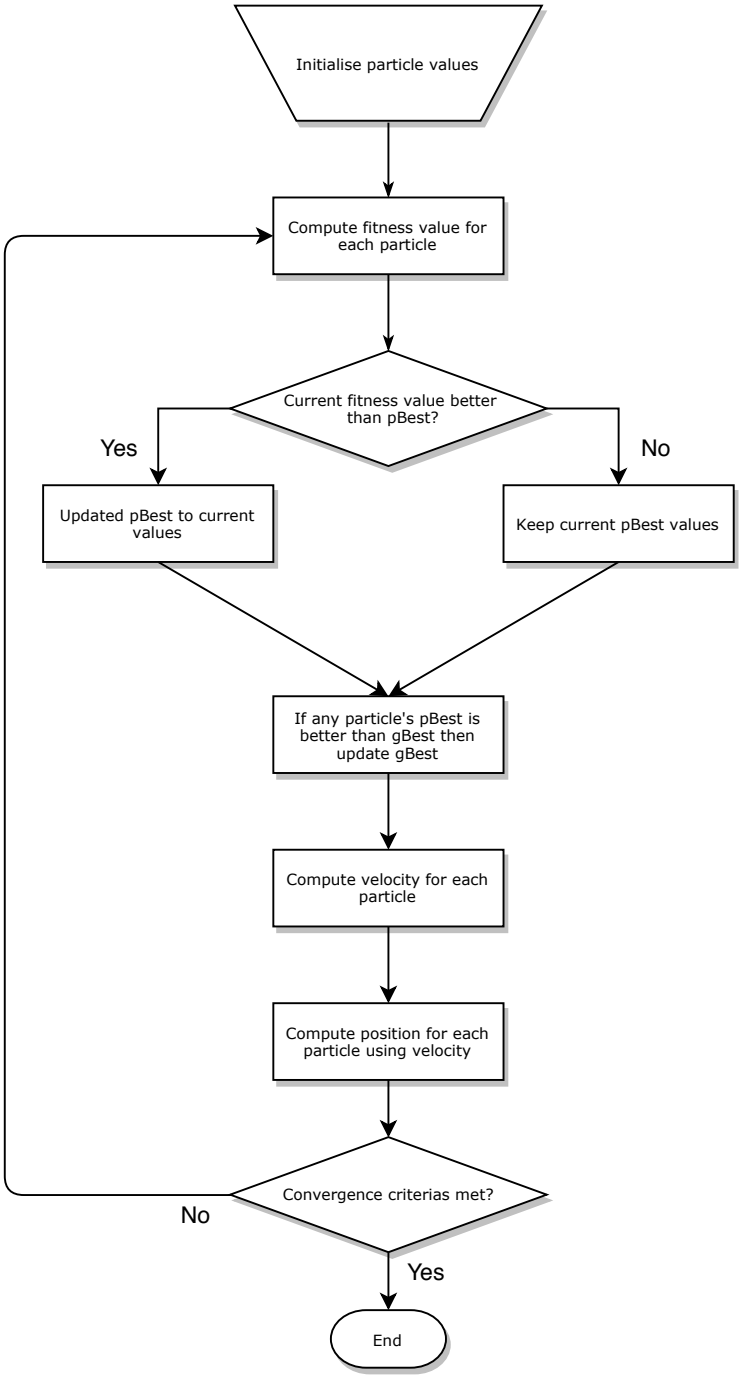


Figure 3.9: Flow chart of PSO algorithm.

Where *pBest* and *gBest* stands for personal best and global best, respectively.

The initialisation of particles is given by:

Algorithm 1 Particle initialisation

```
for each particle do
    for each position do
        p.pos ← var_min + r(var_max − var_min)
```

Where r is a random number $r \in [0, 1]$, var_{min} and var_{max} is the minimum and maximum of the optimisation variable range. Each particle position is stored in $p.pos$.

The velocity update is computed by:

Algorithm 2 Velocity update

```
for each particle do
    for each velocity do
        p.vel ← w(p.vel) + c1q(pBest − p.pos)/Ts + c2r(gBest − p.pos)/Ts
```

Where r and q are random numbers $r, q \in [0, 1]$, w is the inertia weight and c_1 and c_2 are the exploitation and exploration weights, respectively. Each particle velocity is stored in $p.vel$. The inertia weight is updated by a simple rule:

Algorithm 3 Inertia update

```
w ← βw
if w < w_min then
    w ← w_min
```

The position update is the computed by:

Algorithm 4 Position update

```
for each particle do
    for each velocity do
        p.pos ← p.pos + Ts p.vel
        if p.pos > var_max then
            p.pos ← var_max
        if p.pos < var_min then
            p.pos ← var_min
```

A meaningful fitness function needs to be designed in order for the algorithm to measure its results. We are interested in finding candidates for $\boldsymbol{Q}_{lq}$ and $\boldsymbol{R}_{lq}$ which tracks the reference trajectories as close as possible. Thus the fitness function used computes, for each particle in the swarm, a LQ gain matrix K, simulates the closed-loop system using this gain and computes the sum of the Root Mean Square Error between the closed-loop system trajectories and the optimal reference trajectories. The parameters used in the PSO algorithm are as follows:

PSO Parameters	
Parameter	Value
Number of particles	30
Number of variables	8
var_{max}	100
var_{min}	1
w_0	1.4
w_{min}	0.3
c_1	2
c_2	2
β	0.99
T_s	1

Table 3.2: Table showing parameter values for the particle swarm algorithm.

3.6 State estimation

The state estimation is based on the EKF described in 2.4. The main difference is that the measurement functions are all linear, thus removing the need for the Jacobian in the update part of the filter. The car is tracked in x-y position and its orientation using a camera. A gyroscope is used for measuring the angular velocity ω. An accelerometer is used to measure acceleration in x and y. The observer transition function used in the filter is the same as 2.1 but the system has been extended to include acceleration in x and y direction as states.
The transition equation thus becomes:

$$\dot{\mathbf{r}}(t) = \mathbf{f}(\mathbf{r}(t), \mathbf{u}(t)) \tag{3.75}$$

Where:

$$\mathbf{r}(t) = \begin{bmatrix} x(t) & y(t) & \varphi(t) & v_x(t) & v_y(t) & \omega(t) & a_x(t) & a_y(t) \end{bmatrix}^T \tag{3.76a}$$

$$\mathbf{u}(t) = \begin{bmatrix} \delta(t) & \tau(t) \end{bmatrix}^T \tag{3.76b}$$

$$\mathbf{f}(\mathbf{r}(t), \mathbf{u}(t)) = \begin{bmatrix} v_x \cos(\varphi) - v_y \sin(\varphi) \\ v_y \cos(\varphi) + v_x \sin(\varphi) \\ \omega \\ a_x \\ a_y \\ \frac{1}{I_z}(l_f F_{f,y} \cos\delta - l_r F_{r,y}) \\ j_x \\ j_y \end{bmatrix} \tag{3.77}$$

j_x and j_y is the jerk in x and y-direction and directly relates to the control inputs from $\mathbf{u}(t)$.

As the system is sampled in to discrete time the model needs to be rewritten, this is done using modified Euler:

$$\dot{\mathbf{r}}(t+T_s) = \frac{\mathbf{r}(t+T_s) - \mathbf{r}(t)}{T_s}, \quad T_s \to 0 \tag{3.78}$$

$$\mathbf{r}(t+T_s) \approx \mathbf{r}(t) + \mathbf{f}(\mathbf{r}(t), \mathbf{u}(t))T_s = \hat{\mathbf{r}}_{k|k-1} =$$

$$\begin{bmatrix} x + T_s(v_x \cos(\varphi) - v_y \sin(\varphi)) \\ y + T_s(v_y \cos(\varphi) + v_x \sin(\varphi)) \\ \varphi + T_s\omega \\ v_x + T_s a_x \\ v_y + T_s a_y \\ \omega + \frac{T_S}{I_z}(l_f F_{f,y} \cos\delta - l_r F_{r,y}) \\ a_x - a_{x_{(t-T_s)}} + \frac{1}{m}(F_{r,x} + F_{f,x}cos(\delta) - F_{f,y}sin(\delta) + mv_y\omega) \\ a_y - a_{y_{(t-T_s)}} + \frac{1}{m}(F_{r,y} + F_{f,y}cos(\delta) + F_{f,x}sin(\delta) - mv_x\omega) \end{bmatrix} \tag{3.79}$$

Where $a_{x_{(t-T_s)}}$ and $a_{y_{(t-T_s)}}$ are the previous values for the acceleration in x and y direction respectively.
The prediction for the covariance matrix $\boldsymbol{P}^f$ is then calculated from the Jacobian of the transition function, and the process noise described in $\boldsymbol{Q}$ as:

$$\boldsymbol{P}^f_{k|k-1} = \mathbf{F_k}\boldsymbol{P}^f_{k-1|k-1}\mathbf{F}^T_k + \boldsymbol{Q} \tag{3.80}$$

where

$$\mathbf{F}_k = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{r}}\right|_{\hat{r}_{k-1|k-1}, \mathbf{u}_k}, \qquad \boldsymbol{Q} = \begin{bmatrix} 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&\sigma_w&0&0 \\ 0&0&0&0&0&0&\sigma_{ax}&0 \\ 0&0&0&0&0&0&0&\sigma_{ay} \end{bmatrix} T_s \tag{3.81}$$

The update equations 2.17 are calculated separately for each sensor using different measurement functions.

$$\begin{aligned} h_{camera} &= \begin{bmatrix} 1&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0 \end{bmatrix} \\ h_{gyroscope} &= \begin{bmatrix} 0&0&0&0&0&1&0&0 \end{bmatrix} \\ h_{accelerometer} &= \begin{bmatrix} 0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&1 \end{bmatrix} \end{aligned} \tag{3.82}$$

4
Implementation

An attempt to implement the controller for the Kyosho Mini-Z SPORTS AWD was made. The control values for torque and steering angle was intended to be calculated on a PC and then wirelessly transmitted to the Mini-z. The position and orientation measurement was to be implemented on a Raspberry Pi 3 Model B+ running a version of `Linux PREEMPT_RT` and a connected camera and radio transceiver. The Raspberry Pi was intended to be used for camera measurements and communication relay between the control PC and the Mini-z.

The control circuit on the Mini-z was to be removed and a new circuit containing an accelerometer and gyroscope for state estimation, a current sensor measuring the current of the driver motor, two PWM driver circuits for the steering and driver motor, a radio transceiver to transmit sensor data and receive control input for steering angle and driver motor torque, and finally a micro controller for on board signal processing and computations.

The main part of this was implemented in the project however a problem arose. The solver times for the controller turned out to be longer then any reasonable sampling time, even for short a horizons. The quadratic constrains in the optimization problem drastically increased the solution time, but even without the quadratic constraints the solver times were not acceptable. The solver times could of course be reduced using better hardware and a specialised QP solver, however due to the limited resources and time constraints a decision was made to focus on the simulation.

5
Results

In this chapter the results will be presented. A couple of scenarios was been designed for evaluation of the controller. The first scenario will be analysed more in depth than the other scenarios in order to limit the scope of the thesis. As such, the first scenario will feature more data and focus on showing the performance of both the MPCC and the baseline controller. The analyses from the subsequent scenarios will therefor focus more on the MPCC and the driven trajectories. Important to note is that the results presented in this chapter should not be viewed as a comparison on which controller is better. The MPCC is essentially a path follower scheme while the baseline controller is a trajectory tracking scheme, where the trajectory is offline calculated. The results should instead be seen as how close the MPCC can come to time optimality whilst using a moderate finite horizon and as well retaining its stability guarantees. The weighting matrices for the baseline controller can be found in Appendix C.
Below is a table of the default parameters used in both simulation and implementation, unless other wise stated:

Default parameter values

Parameter	Value
N	40
q_l	250
q_c	0.025
q_θ	0.25
q_ϵ	1000
q_{ϵ^2}	1000
r_δ	1
r_τ	1
r_θ	1
α	0.99
α_{min}	-0.2 rad
α_{min}	0.2 rad
v_{max}	3 m/s (without obstacle)
v_{max}	2 m/s (with obstacle)

Table 5.1: Table showing the default parameter values for the Model Predictive Contouring Controller.

Note that in the following control trajectory plots the control input τ has been scaled

with a factor of 100 for better readability.

5.1 Scenario 1: S-curve

In this scenario, both the MPCC and the baseline controller will perform an s-curve manoeuvre. We will analyse the data and showcase the strength and similarities between the controllers. The car will start with maximum allowed velocity, as we are looking to see how the s-curve will be navigated in a high speed manner. Thus the initial conditions for this scenario is as follows:

$$\boldsymbol{\xi}_0 = [0 \quad 0 \quad 0 \quad 2 \quad 0 \quad 0]$$

Final conditions for the Time optimal trajectory computation were given as:

$$\boldsymbol{x}_{tf} = [2.3 \quad free \quad free \quad free \quad free \quad free]$$

5.1.1 Model Predictive Contouring Controller

In this section we will show the results from the Model Predictive Contouring Controller on the s-curve scenario. In Figure 5.1 the predicted trajectories for time $k+i$ and the actual trajectory for time k are shown.

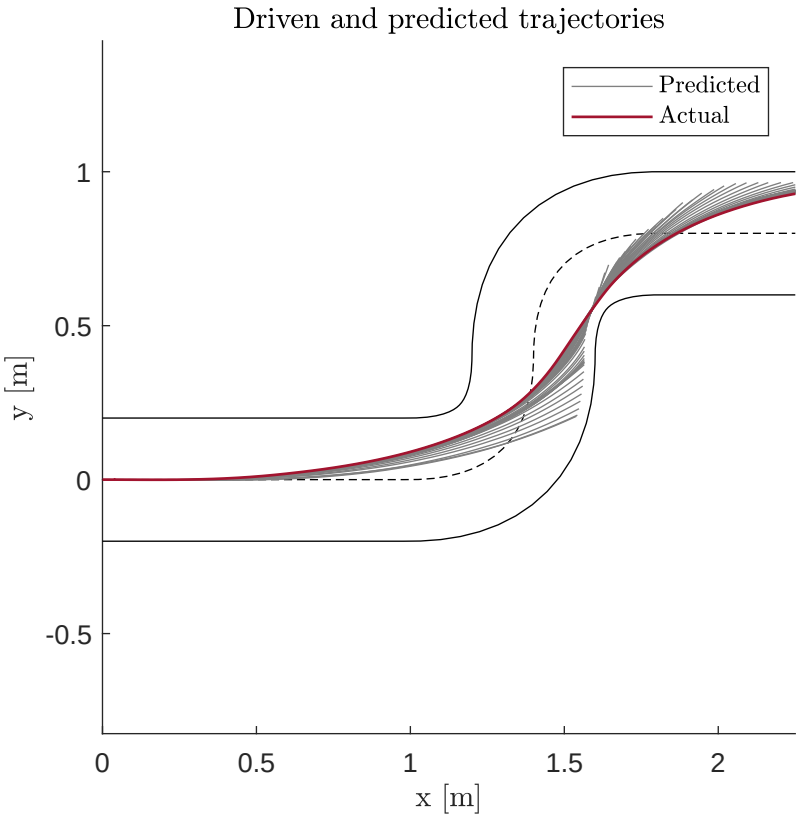


Figure 5.1: Driven and predicted trajectories.

The predicted trajectories shows the evolution of how the controller navigates the given scenario. It is clear that due to the finite horizon the controller cannot foresee the second curve until roughly one third into the scenario. The S-curve scenario shows that the Contouring-scheme works as intended, meaning that the car "cuts" the corners due to the maximization of progress of θ_k on the centre line.

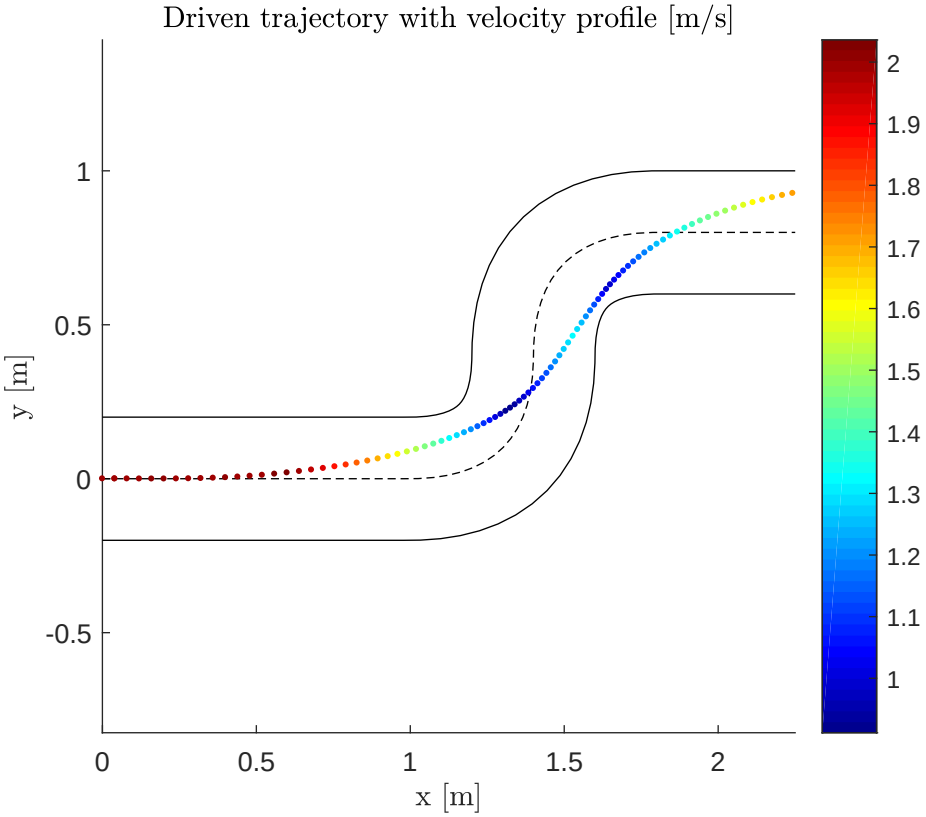
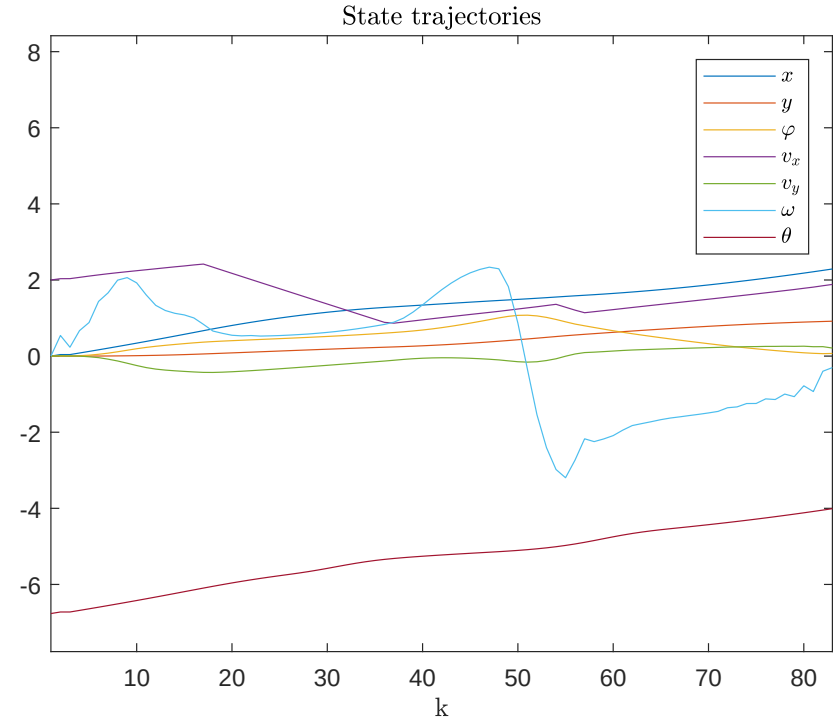
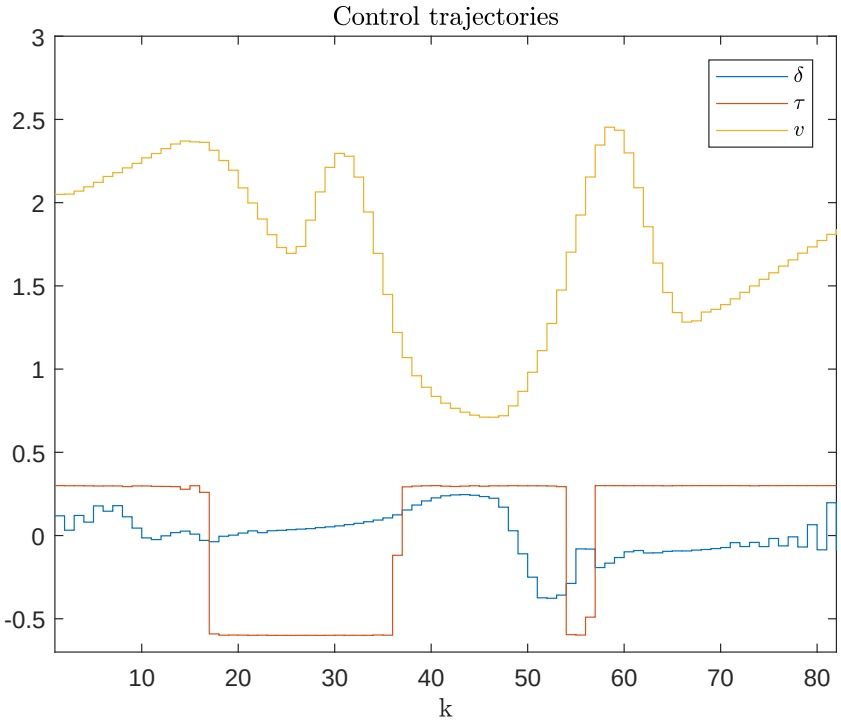


Figure 5.2: LTVMPCC driven trajectory for Scenario 1.

The velocity profile of the actual driven trajectory in Figure 5.2 shows how the controller slows the car down in order to clear the corners.



(a) State trajectories.



(b) Control trajectories.

Figure 5.3: LTVMPCC state and control trajectories for Scenario 1.

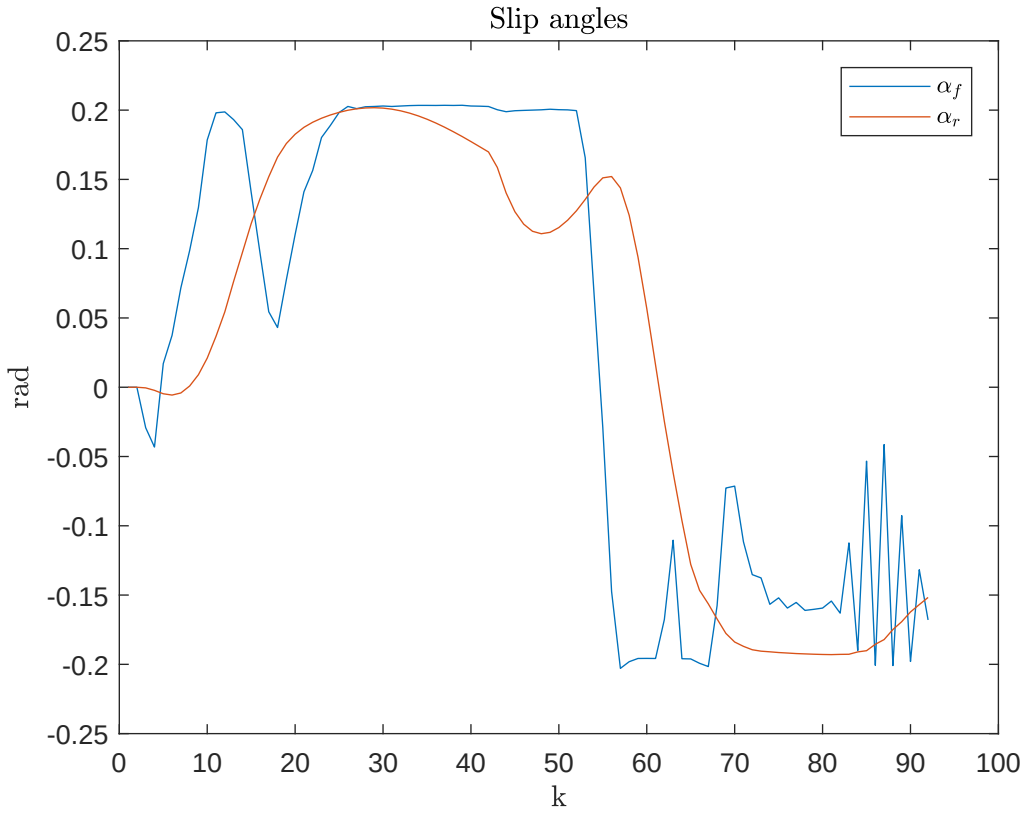


Figure 5.4: LTVMPCC slip angles for Scenario 1.

The deceleration can be seen as well in the control trajectories shown in Figure 5.3b, the controller applies full braking as the car nears the first corner. In Figure 5.4 we can see that the slip angle constraints are hitting their respective limits for both corners. Examining further it is clear that the first corner is especially difficult. The forward slip angle constraint α_f is on its limit as the car is approaching, and navigating the first corner. This translates to what is known as under-steer, meaning the front steering wheels are not generating the lateral force needed to turn the car due to wheel slip. In summary, the LTVMPCC manages to control the car through a difficult S-curve where the car is experiencing severe slip due to the high initial velocity.

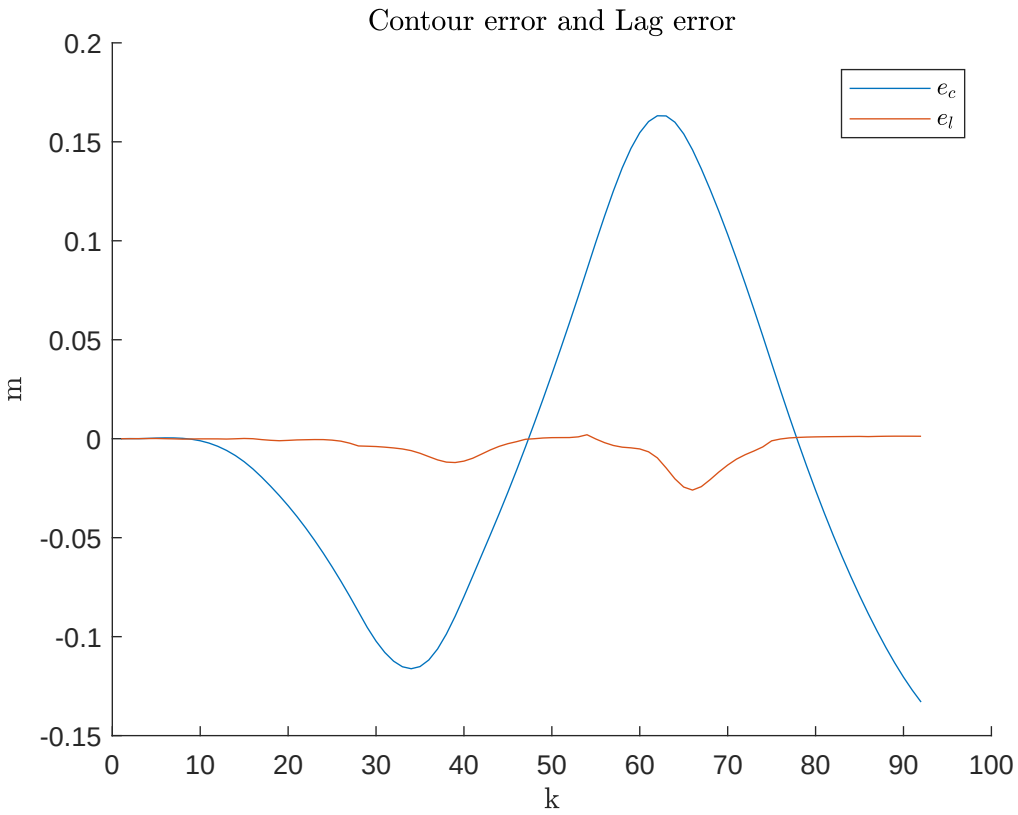


Figure 5.5: Contouring error and lag error for Scenario 1

Figure 5.5 shows how the controller manages to keep the lag error e_l low while allowing the contour error e_c to grow when needed.

Values of $\bar{\alpha}$	
Time step	$\bar{\alpha}$
$k = 40$	0.942147594081742
$k = 80$	0.99
$k = 120$	0.716791479087716

Table 5.2: Table showing values of $\bar{\alpha}$ for Scenario 1.

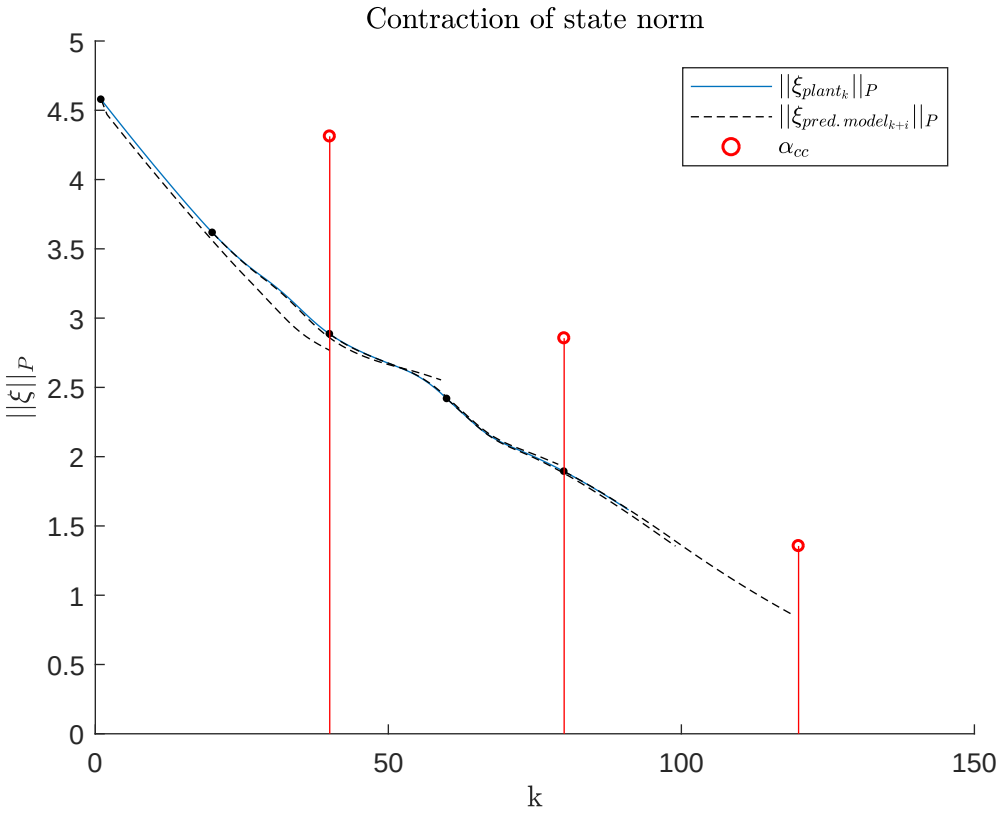


Figure 5.6: Contraction of state norm for Scenario 1.

In Figure 5.6 we see how the norm of the state is decreasing over the course of the S-curve scenario. The norm of the state is well below the upper bound $\bar{\alpha}$ at each contraction step. The computed values of $\bar{\alpha}$ during the simulation were as follows: At time $k = 40$ the computed upper bound $\bar{\alpha}$ is not very restrictive. This is due to the fact that it is computed at time $k = 0$, at which the predicted trajectories have not yet begun to adjust much for the coming corner section. Thus at the time of computing the bound, the car is driving in relatively steady state meaning there is little model/plant mismatch. At time $k = 80$ no feasible bound could be computed due to not finding a Lipschitz constant which satisfy the assumptions stated in 3.3.4 over the prediction trajectories at time $k = 40$. The last bound computed at time $k = 120$ is quite restrictive. Accelerating out of the second corner and on to the straight lane while having saturated lateral forces could yield significant model/plant mismatch. Hence the contraction constraint enforces a strong contraction of the

state norm to ensure stability.

5.1.2 Baseline controller

In this section we will show the results from the baseline Linear Quadratic controller. The time optimal driven trajectory which is calculated offline is also presented.

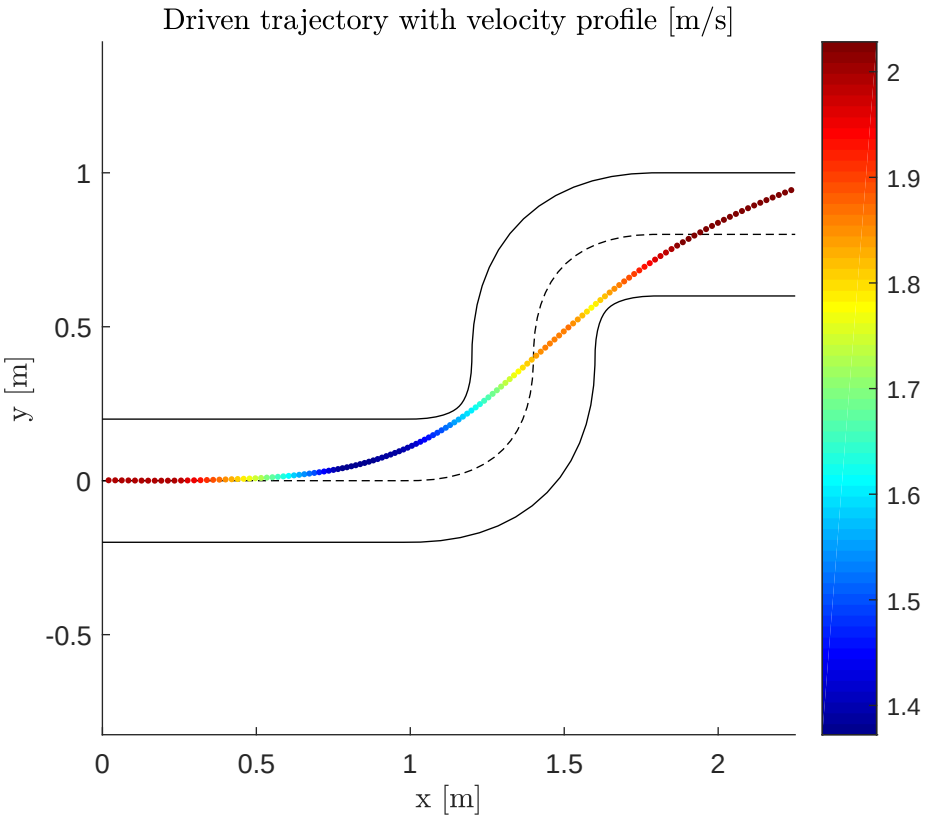
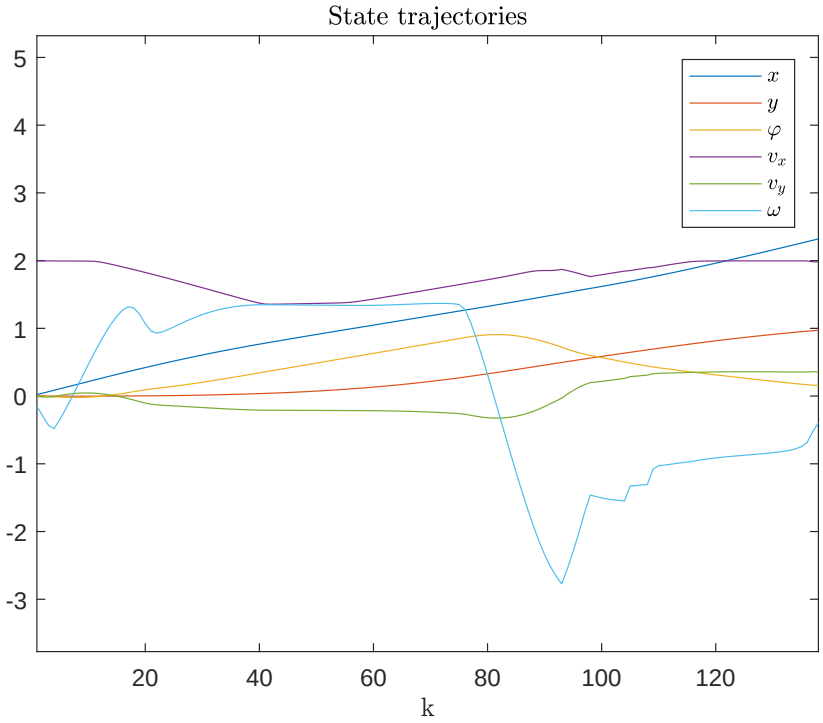
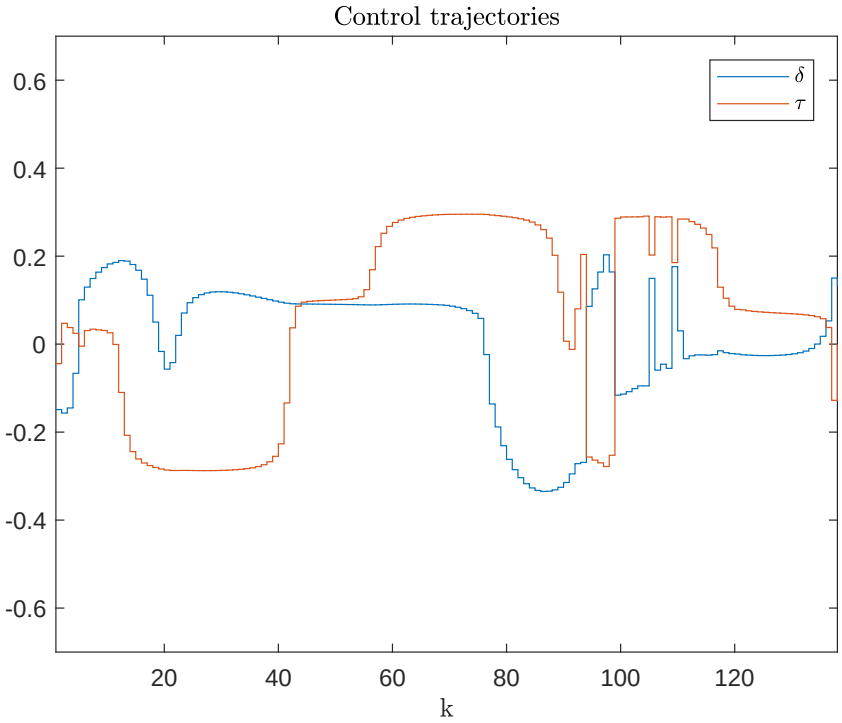


Figure 5.7: LQ driven trajectory for Scenario 1.

Figure 5.7 shows the driven trajectory using the LQ controller. The controller is using the offline calculated time optimal trajectory as reference. Thus we see how the velocity is kept high during the S-curve in contrast to our Model Predictive Contouring Controller which does not have full knowledge of what is coming up.



(a) State trajectories.



(b) Control trajectories.

Figure 5.8: LQ state and control trajectories for Scenario 1.

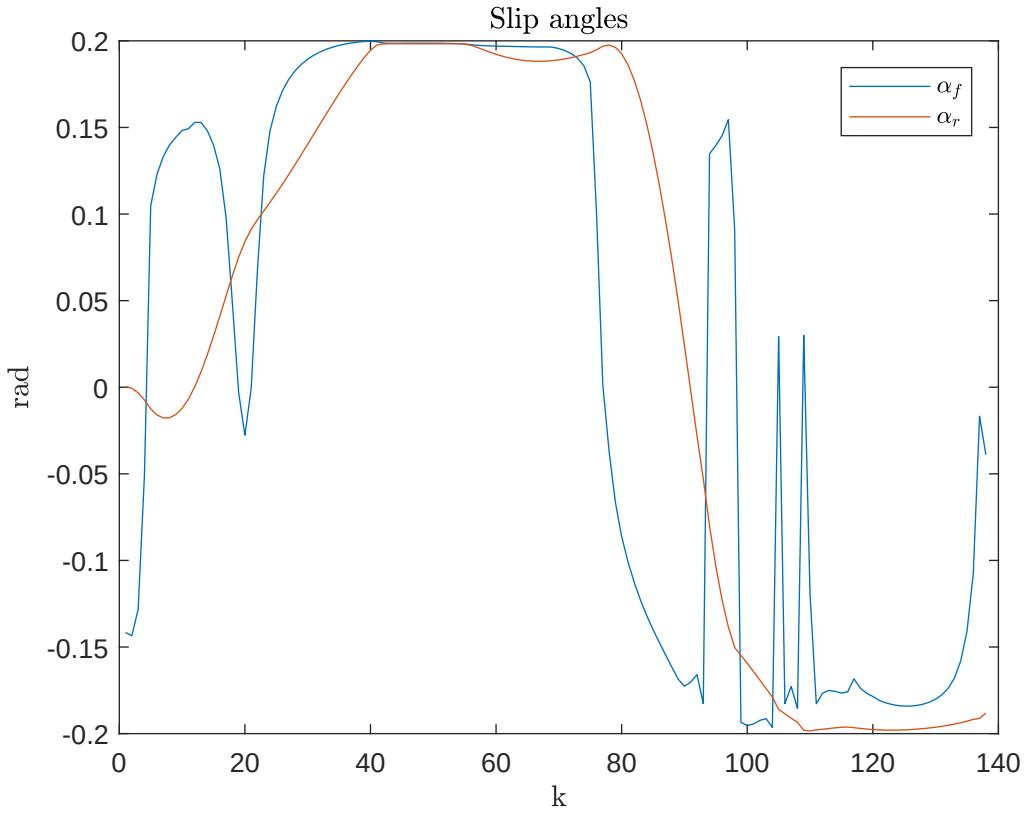


Figure 5.9: LQ slip angles for Scenario 1.

Just as with our main controller the slip angles constraints are on their limits for a good part of the simulation time which can be seen in Figure 5.9. This shows that due to the high initial velocity even the time optimal trajectory consists of braking hard and steering into the initial corner, causing under-steer. The state and control trajectories in Figure 5.8 confirms this.

5.1.3 Comparison

In this section a comparison plot is shown and further data will be presented for comparison purposes.

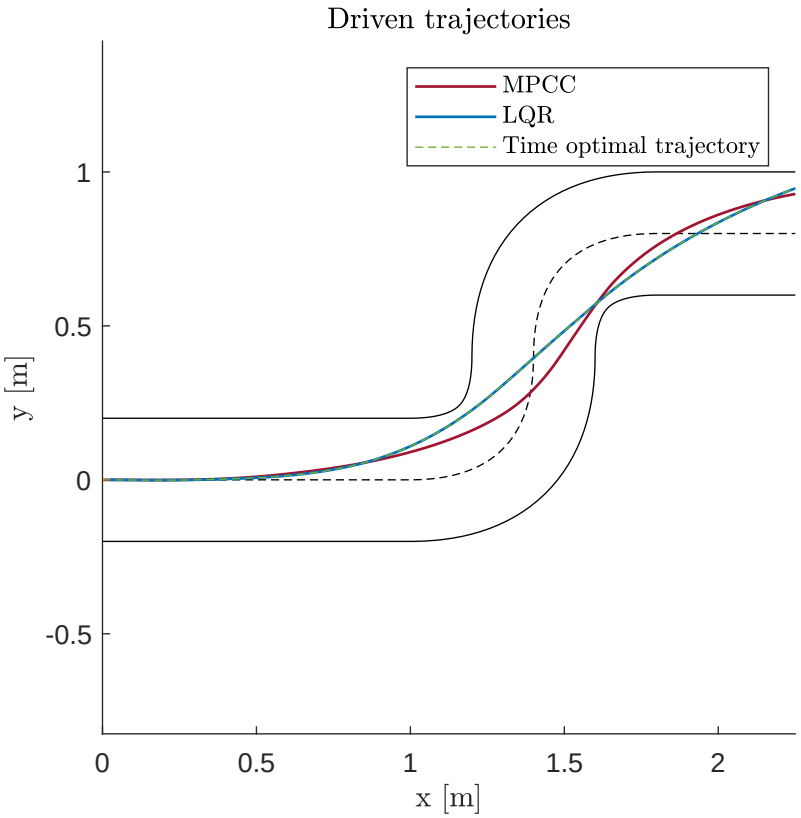


Figure 5.10: Driven trajectories for Scenario 1.

In Figure 5.10 we see the Model Predictive Contouring Controller trajectory, the LQ trajectory and the Time optimal trajectory superimposed. The final time for the MPCC was $t_f = 1.82$ seconds. For the Time optimal trajectory/LQ Controller the final time was $t_f = 1.48$ seconds. This makes the trajectory yielded by the LQ controller roughly 18.7% faster than the one yielded by the MPCC.

5.2 Scenario 2: S-curve with obstacle

In this scenario we present the same S-curve as in the previous scenario, however with an added an obstacle in a rather inconvenient position on the track. The obstacle is placed at $x, y = \{1.45, 0.4\}$. This position of the obstacle will force both the MPCC trajectory and the Time optimal trajectory to change with respect to their trajectories from Scenario 1. Initial conditions and final conditions from Scenario 1 are repeated as well.

5.2.1 Model Predictive Contouring Controller

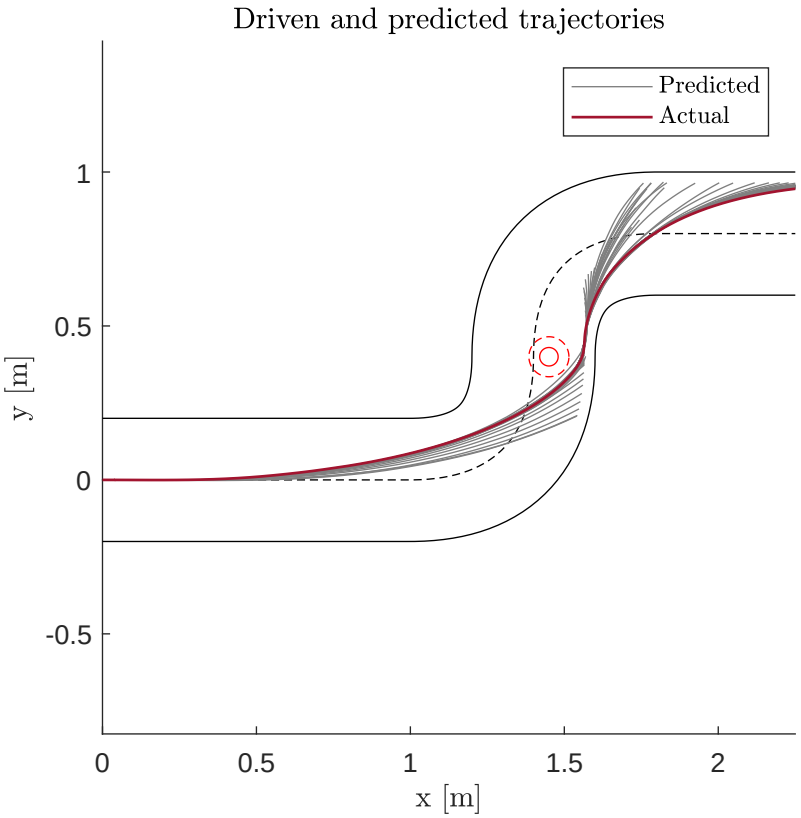


Figure 5.11: Driven and predicted trajectories.

The addition of the obstacle requires the path planner to modify the lane constraints. As can be seen in Figure 5.11 the path planner modifies the constraint such that the car can pass to the right of the obstacle. This choice is a result of minimising the energy spent of passing the obstacle.

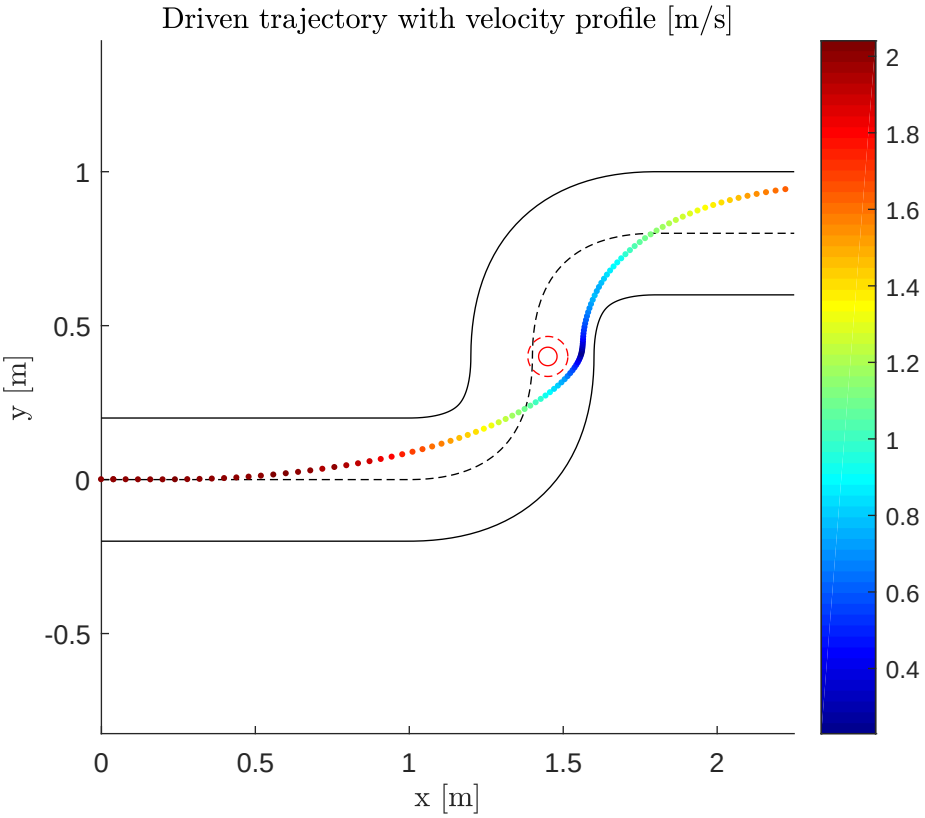
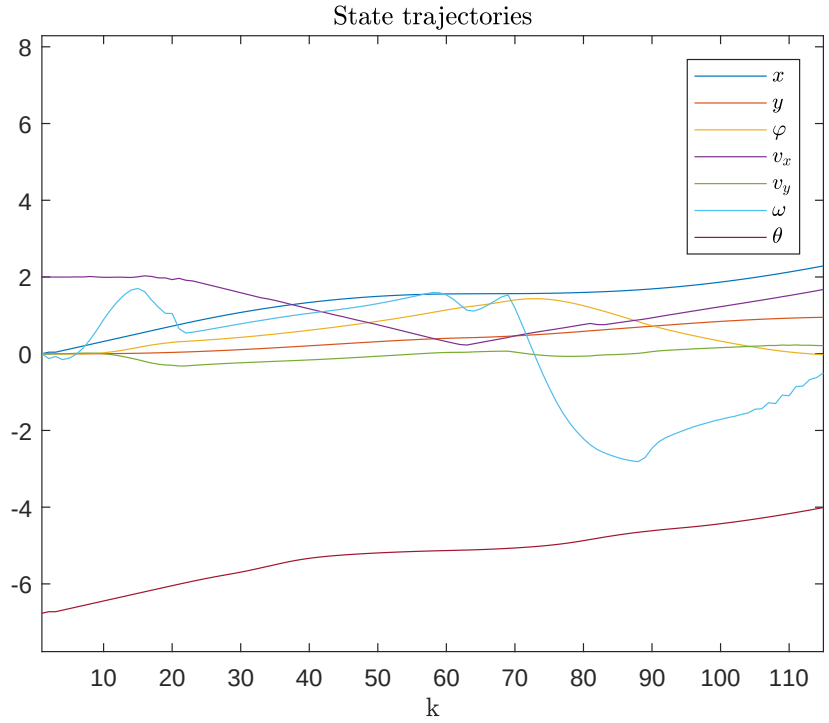
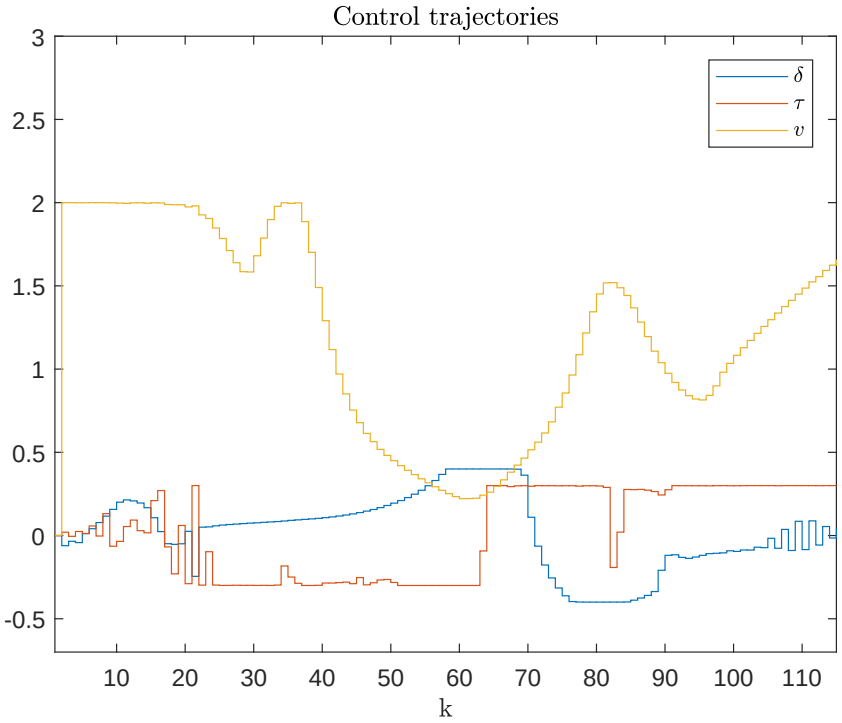


Figure 5.12: LTVMPCC driven trajectory for Scenario 2.

The velocity profile in Figure 5.12 shows a significant drop in velocity when passing the obstacle. The torque input trajectory in Figure 5.13b show how the controller is applying full braking for a large period of time while approaching the corner with the obstacle. Navigating around this obstacle at such large initial velocity is a difficult task. Despite this the controller still manages to keep the slip angles within their limits (as posed, using predicted longitudinal velocity) as can be seen in Figure 5.14.



(a) State trajectories.



(b) Control trajectories.

Figure 5.13: LTVMPCC state and control trajectories for Scenario 2.

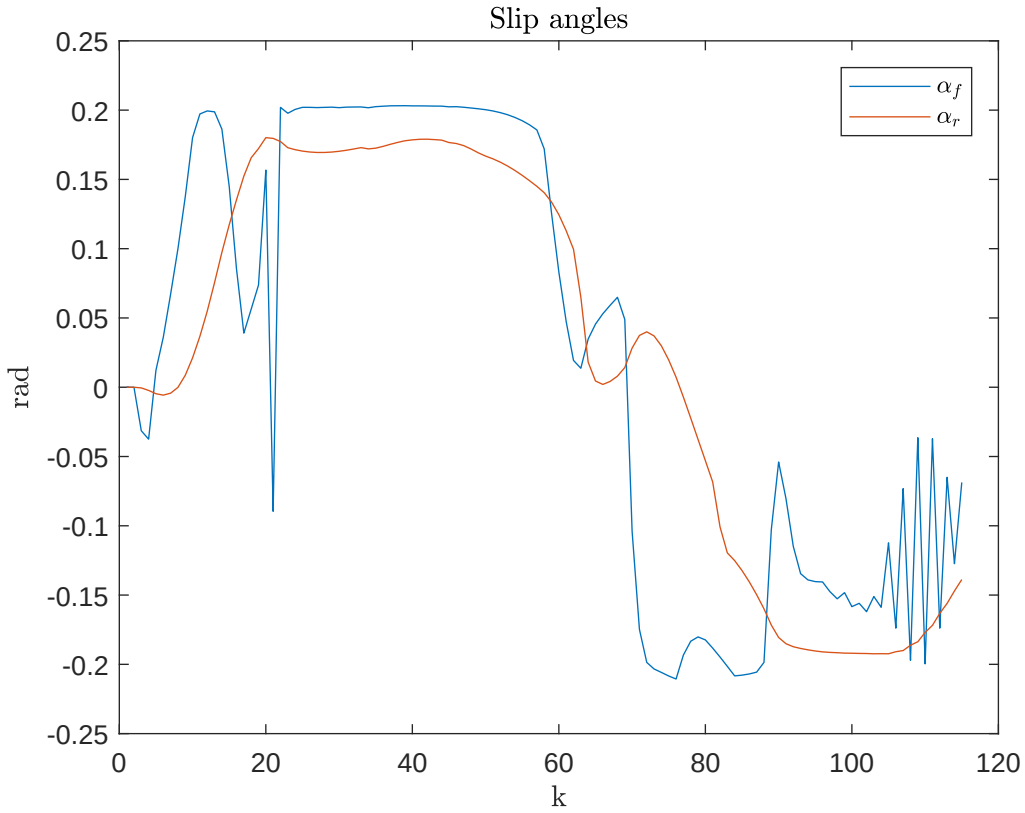


Figure 5.14: LTVMPCC slip angles for Scenario 2.

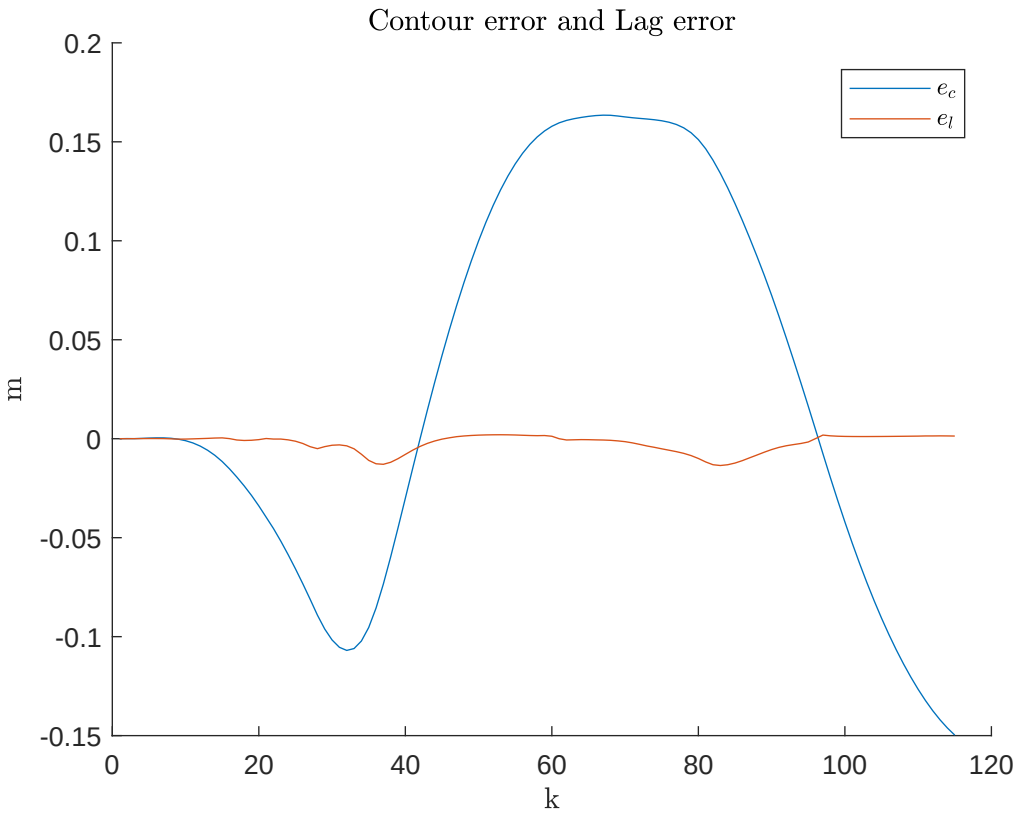


Figure 5.15: Contouring error and lag error for Scenario 2.

Figure 5.15 shows the contour and lag error. It can be seen that the important lag error is kept small while allowing the contour error to grow as necessary.

Values of $\bar{\alpha}$	
Time step	$\bar{\alpha}$
$k = 40$	0.943061862496222
$k = 80$	0.99
$k = 120$	0.99

Table 5.3: Table showing values of $\bar{\alpha}$ for Scenario 2.

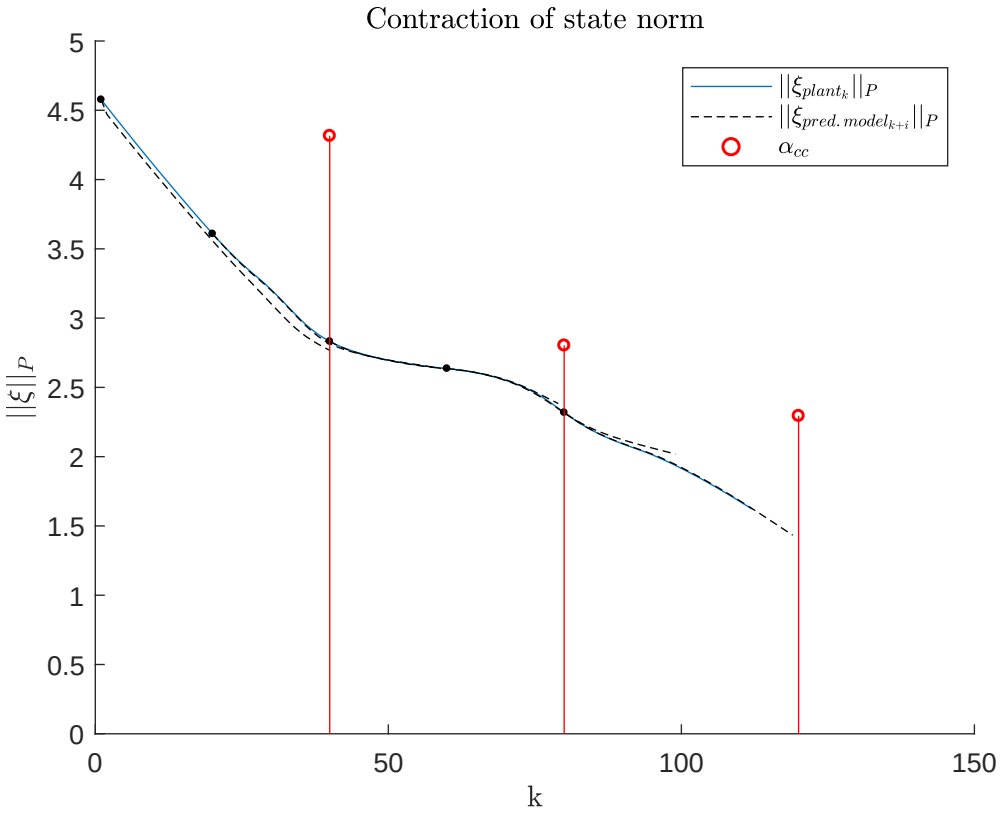


Figure 5.16: Contraction of state norm for Scenario 2.

Figure 5.16 shows the contraction of the state norm during the scenario. In contrast to the contraction of the state norm in scenario 1 we can here see that between contraction step one and two, i.e $j = 1 \longrightarrow j = 2$ the contraction of the state norm has almost stagnated. This is due to the car moving slowly around the obstacle thus the path variable θ is not decreasing as much as in scenario one. Given that the contraction constraint is constructed with the weighted norm, with emphasis on θ this is expected. However, the state norm is not required to be decreasing between time steps k, only between contraction steps j, thus stability is still ensured.
The table shows the computed values of $\bar{\alpha}$. Feasible values for contraction steps $j = 2$ and $j = 3$ could not be computed due to not finding a small enough Lipschitz constant.

5.2.2 Comparison

In this section a comparison plot is shown and further data will be presented for comparison purposes.

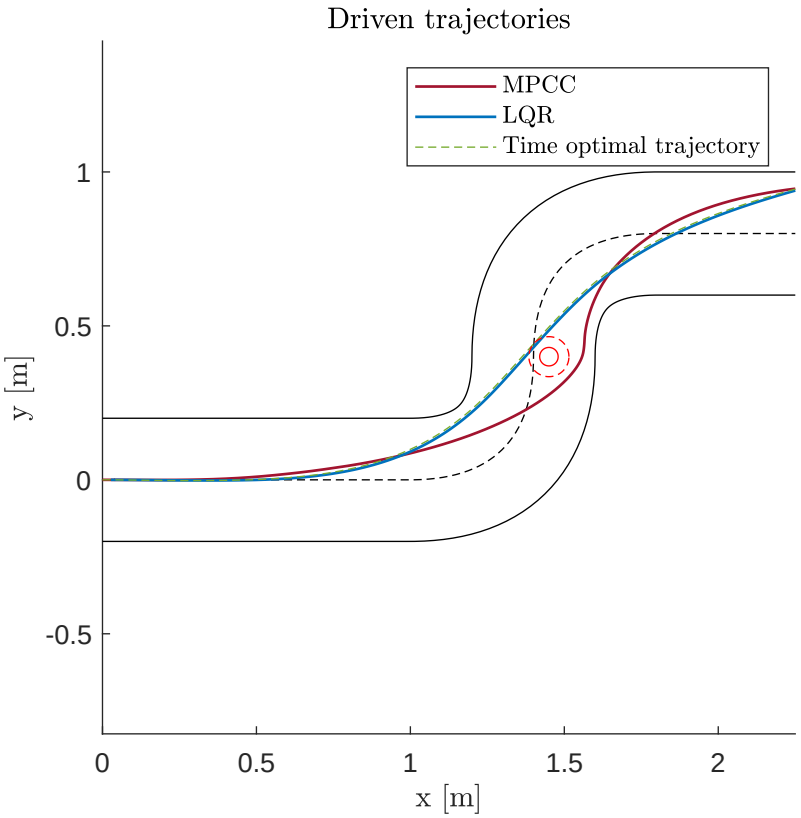


Figure 5.17: Driven trajectories. for Scenario 2.

Figure 5.17 shows the trajectories driven by both controllers. As can be seen the Time optimal trajectory is passing to the left of the obstacle. This yields a faster trajectory. The finite horizon of the MPCC limits its decision making and as such the path planner (which only uses the same finite horizon information as the controller) decides to pass to the right of the obstacle. The final time for the Model Predictive Contouring Controller was $t_f = 2.28$ seconds, while the final time for the baseline LQ controller was $t_f = 1.88$ seconds. Making the trajectory yielded by the LQ controller roughly 17.5% faster. However, important to note that LQ controller cannot track the time optimal trajectory well enough and thus crashes into the obstacle.

5.3 Scenario 3: Straight road with obstacle

In this scenario an obstacle is placed in the middle of a straight road. The goal for the controller is to steer clear of the obstacle and then return to the centre line. This manoeuvre is called a double lane change manoeuvre and is often found in other literature related to automotive control, for example in [8]. The obstacle is placed

at $x, y = \{2, -0.001\}$. The position is slightly perturbed south from $y = 0$ in order to avoid dual solutions.

5.3.1 Model Predictive Contouring Controller

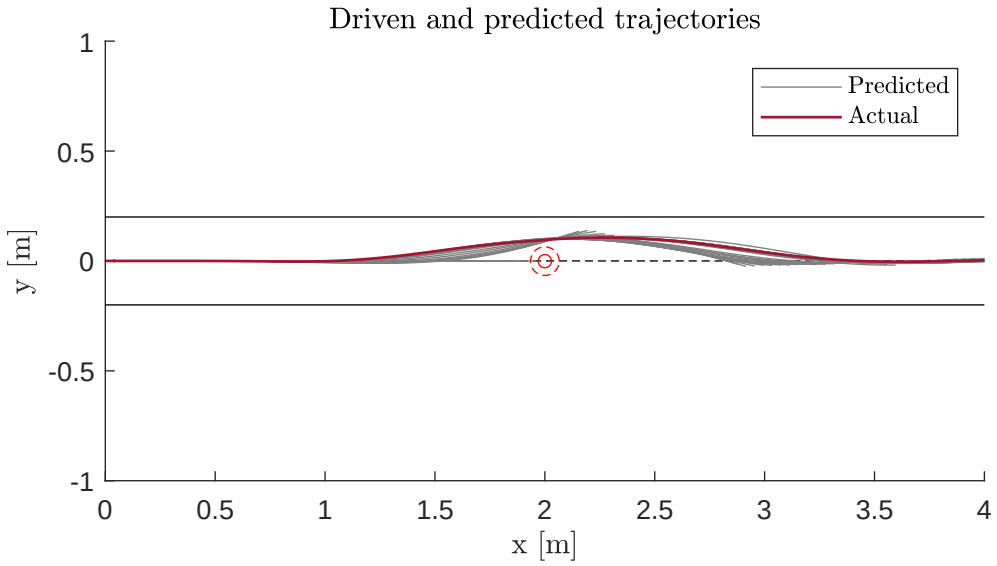


Figure 5.18: Driven and predicted trajectories.

Figure 5.18 shows the driven and predicted trajectories. The controller steers clear of the obstacle and back to the centre line in reasonable amount of space.

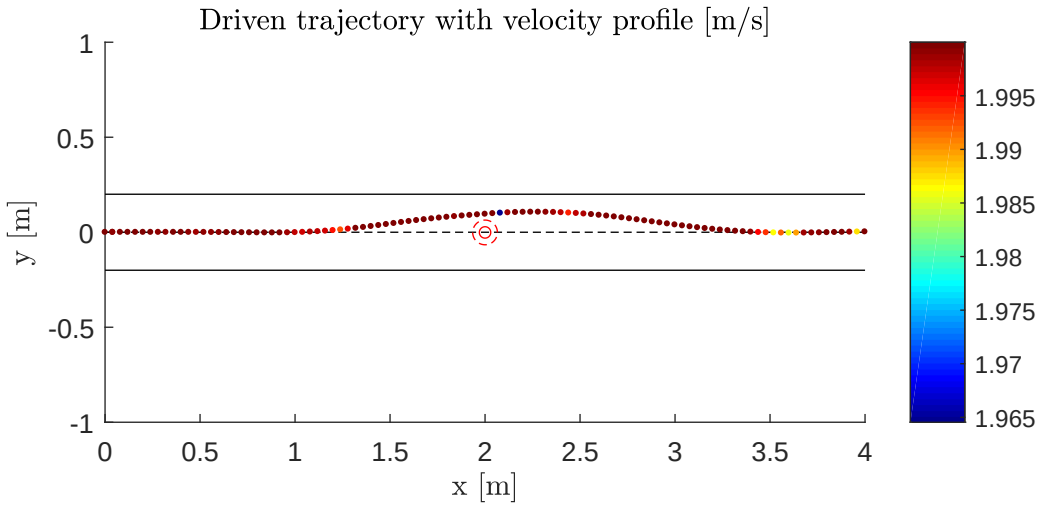
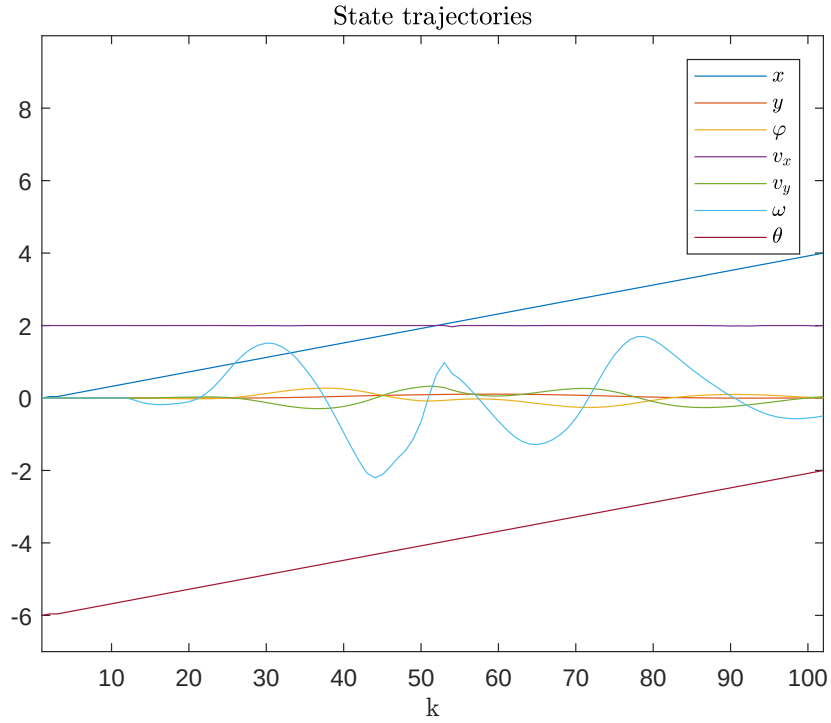
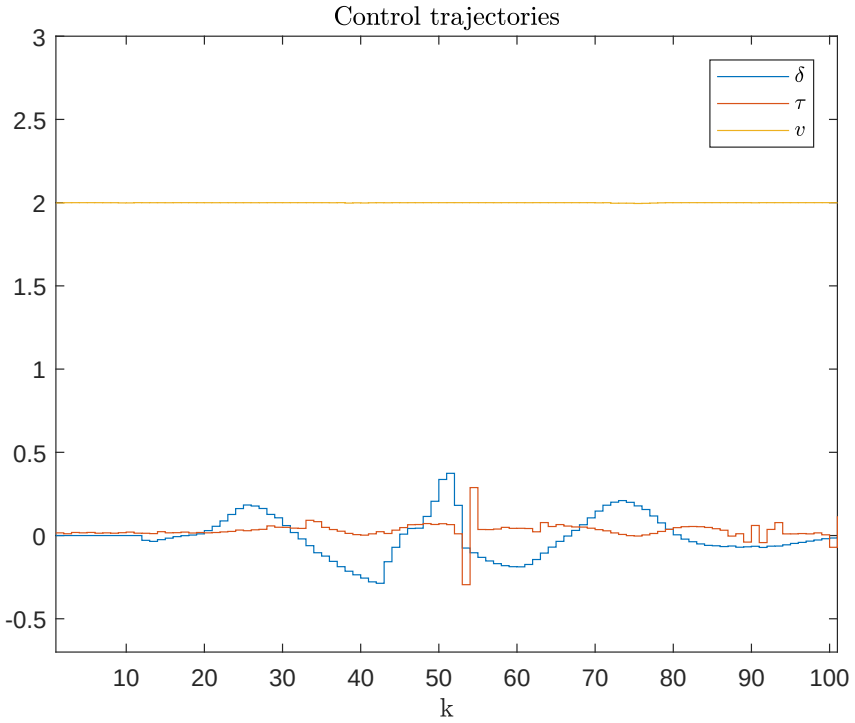


Figure 5.19: LTVMPCC driven trajectory for Scenario 3.

Figure 5.19 shows the driven trajectory along with its velocity profile. It can be seen that the controller kept an almost constant max velocity over the scenario. Only slowing down, relatively, as the car is approaching the centre line again after having passed the obstacle and finding a straight trajectory towards the end.



(a) State trajectories.



(b) Control trajectories.

Figure 5.20: LTVMPCC state and control trajectories for Scenario 3.

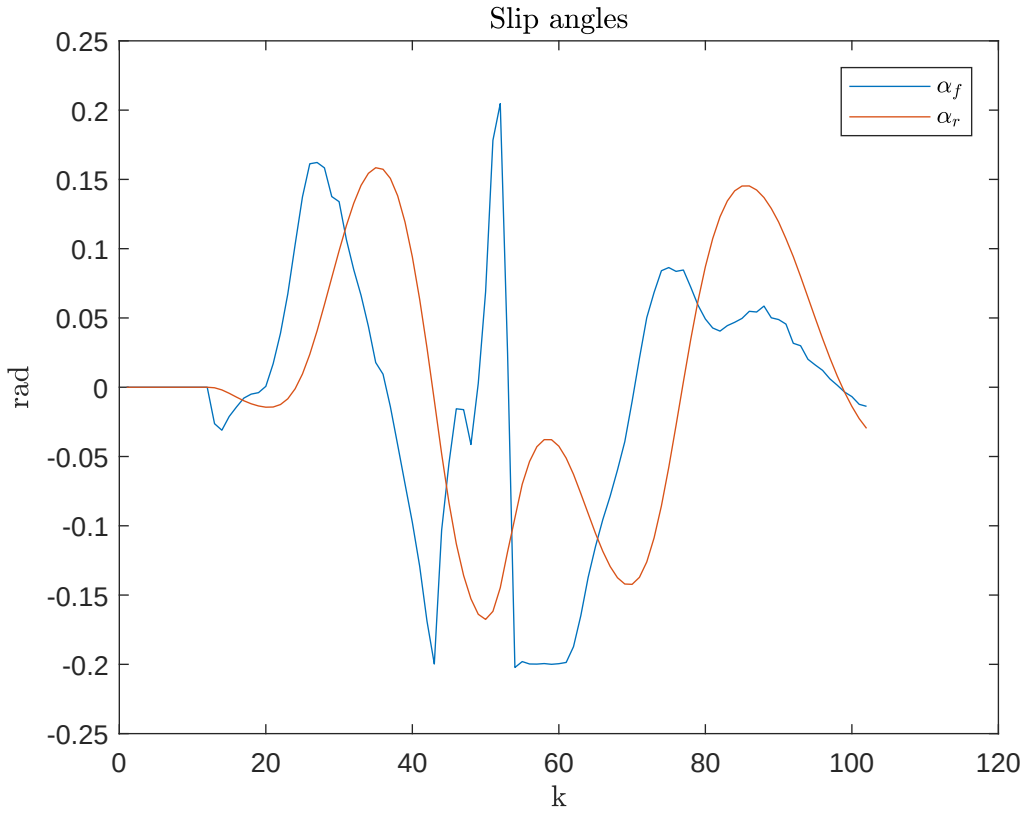


Figure 5.21: LTVMPCC slip angles for Scenario 3.

Figure 5.20 shows the state and control trajectories over the scenario. The controller struggled to keep the volatility of the heading low. Thus the oscillatory trajectories seen in heading, rate of change of heading and steering input. This is due to lack of grip at the high velocity the car is travelling, resulting in understeer, oversteer and the controller then counter-steering to bring the sliding rear end of the car back into line with the heading. In Figure 5.21 the same oscillatory behaviour is naturally present. The offset between α_f and α_r shows understeer followed by oversteer. However, despite the lack of grip the controller manages to steer the car around the obstacle, at almost max velocity, while still keeping the slip angles within its limits.

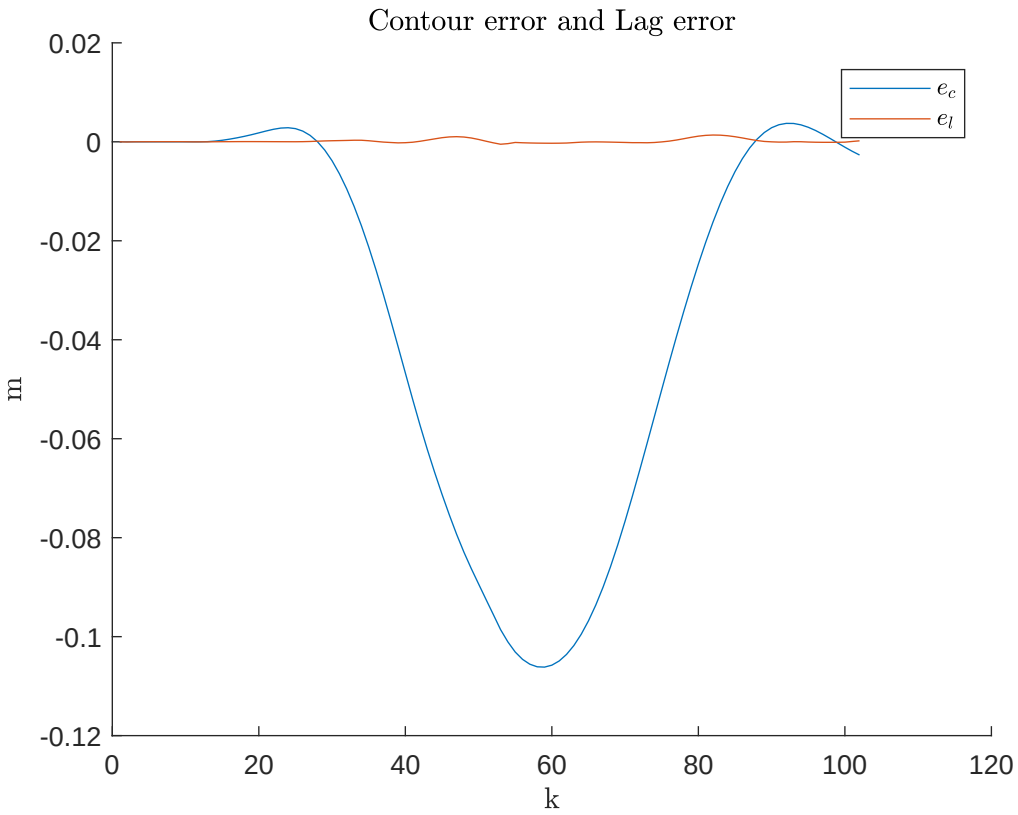


Figure 5.22: Contouring error and Lag error for Scenario 3.

In Figure 5.22 it can bee seen that the controller keeps the lag error e_l at almost zero over the course of the scenario, while letting the contour error e_c grow in order to pass the obstacle.

Values of $\bar{\alpha}$	
Time step	$\bar{\alpha}$
$k = 40$	0.999899868706781
$k = 80$	0.886181011494793
$k = 120$	0.994086242762391

Table 5.4: Table showing values of $\bar{\alpha}$ for Scenario 3.

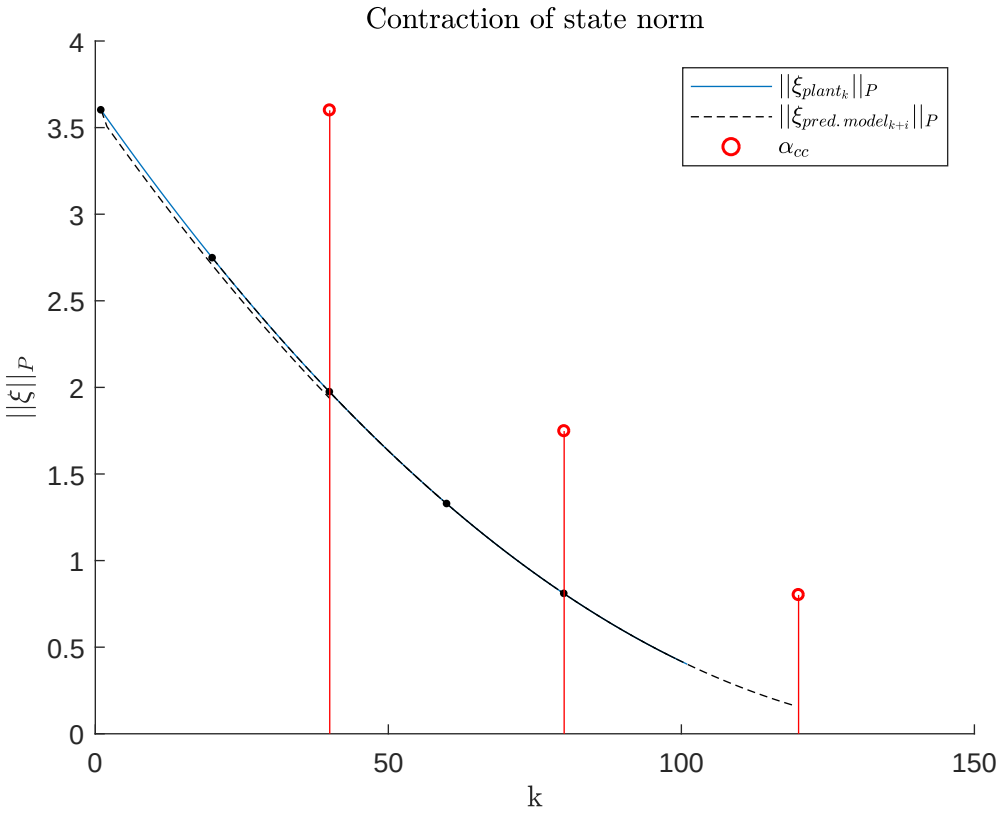


Figure 5.23: Contraction of state norm for Scenario 3.

Figure 5.23 shows the contraction of the state norm during the scenario. The controller had no difficulties in decreasing the norm as the path variable θ could be maximised with great efficiency.
Upper bounds $\bar{\alpha}$ could be computed at each contraction steps. Where the upper bound computed at $k = 80$, $j = 2$ is significantly more restrictive. This is due to the car manoeuvring around the obstacle at high speed giving the need for a more restrictive bound as the model/plant mismatch increases.

5.3.2 Comparison

In this section a comparison plot is shown and further data will be presented for comparison purposes.

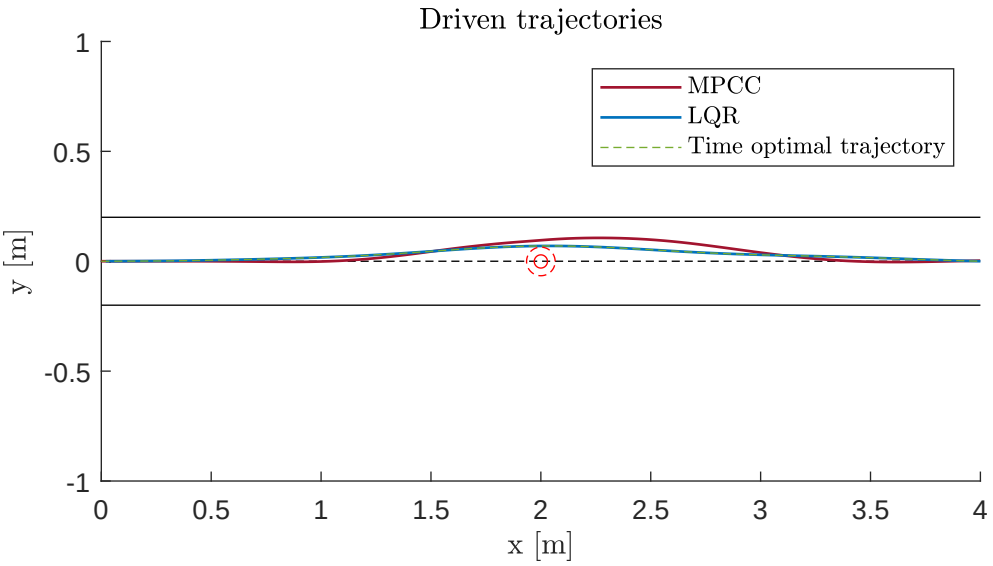


Figure 5.24: Driven trajectories for Scenario 3.

Figure 5.24 shows both drive trajectories. The trajectory yielded by the Time optimal optimisation problem is tracked well enough by the LQ controller. The trajectory given by the LQ controller passes the obstacle more close than the trajectory yielded by the LTVMPCC. However the latter is quicker to return to the centre line. The final time yielded by the LQ controller was $t_f = 2.03$ while the final time yielded by the LTVMPCC was $t_f = 2.04$. Making the trajectory yielded by the LQ controller roughly 0.5% faster than the one by the LTVMPCC.

5.4 Scenario 4: Track with and without added output noise

In this scenario we show the results of driving one lap around the course. Firstly using the MPCC with full state feedback, then secondly the MPCC using estimated state feedback where a plant/model mismatch has been introduced as well.
For this section some plots which are present in previous sections has been omitted in order to limit the space of the thesis.

5.4.1 MPCC

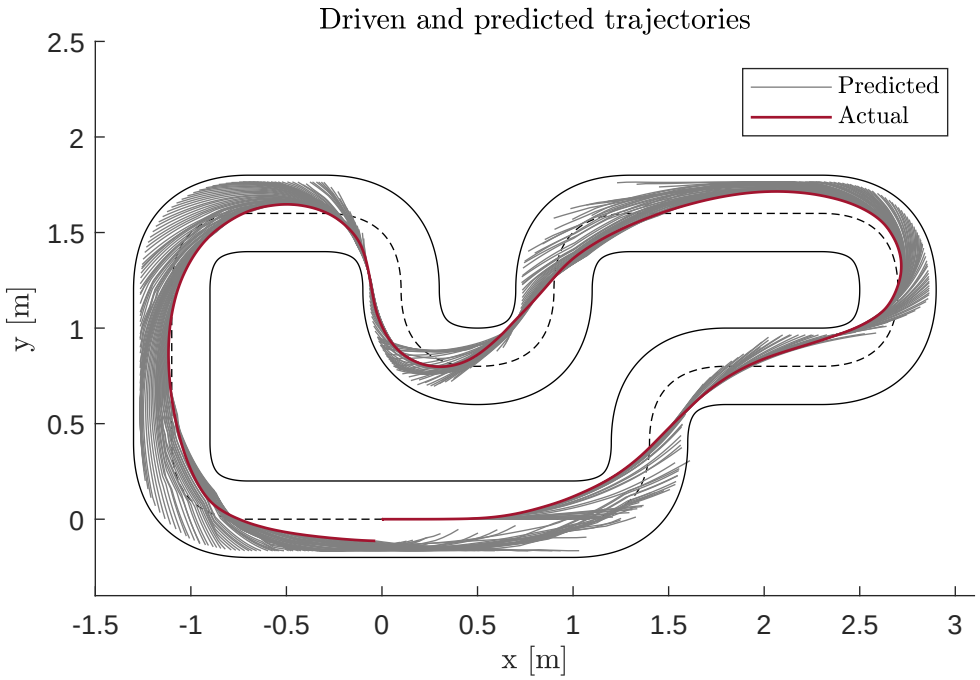


Figure 5.25: LTVMPCC driven and predicted trajectories for Scenario 4.

Figure 5.25 shows the driven and predicted trajectories during one lap of the track. The controller kept a racing line through most of the track. The controller had some slight difficulties in navigating the u-turn seen at the top of the track. Due to the finite horizon the controller does not know the car is actually entering a u-turn segment, as can be seen by the predicted trajectories. Thus it enters this segment with a high velocity, causing the controller to take a very wide turn when exiting the u-turn segment.

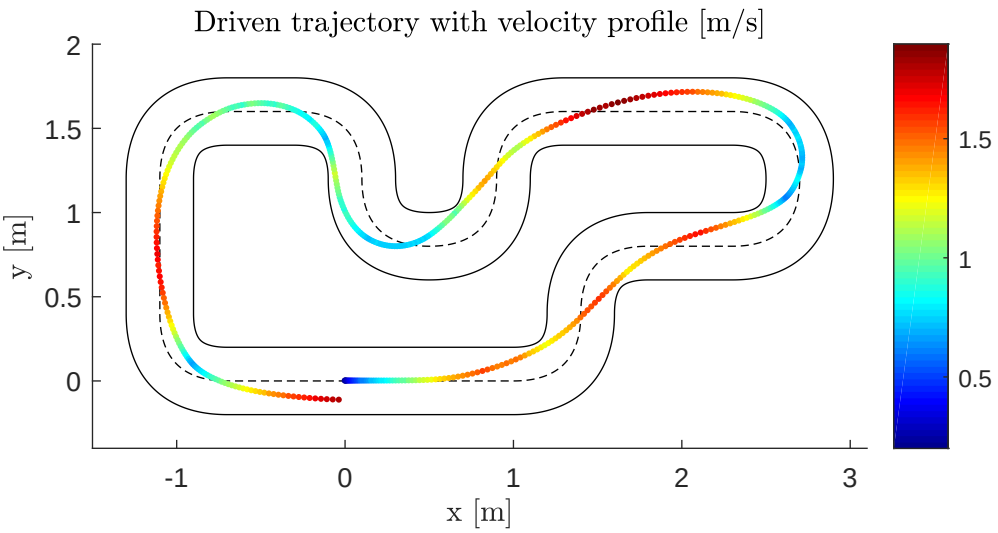
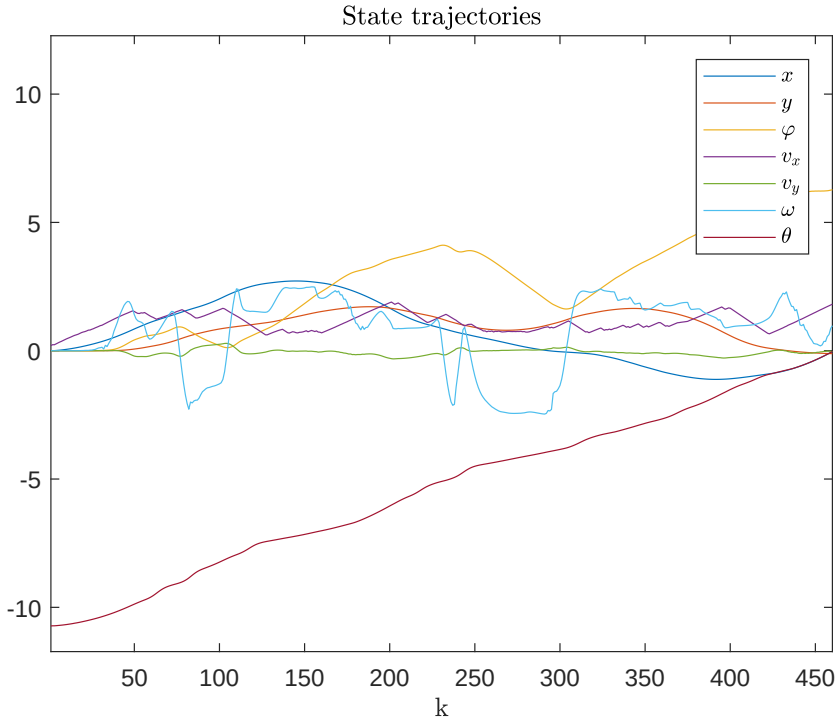
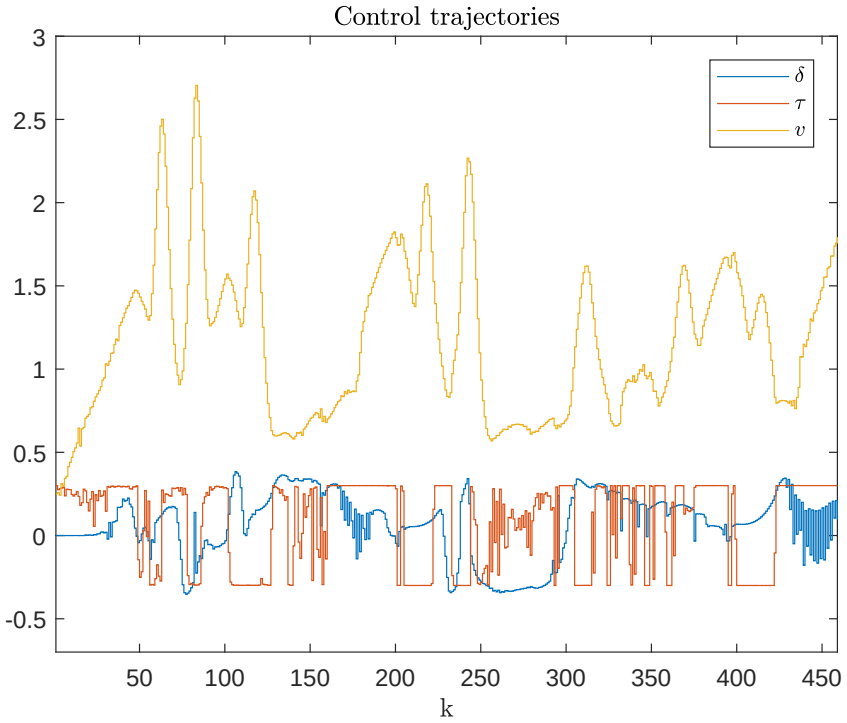


Figure 5.26: LTVMPCC driven trajectory for Scenario 4.

Figure 5.26 shows the driven trajectory along with its velocity profile. As one would expect, the controller kept the car at high velocity during the the straight lines, slowing down when approaching curves, and speeding up when exiting curves.



(a) State trajectories.



(b) Control trajectories.

Figure 5.27: LTVMPCC state and control trajectories for Scenario 4.

Figure 5.27 shows the state and control trajectories for the scenario. One can see that the path variable θ_k is driven to zero and that the virtual input v, which drives θ_k, varies a great deal during the lap. This shows the strength of letting the controller decide the evolution of the path variable rather than have it predefined, as in a reference tracking setup.

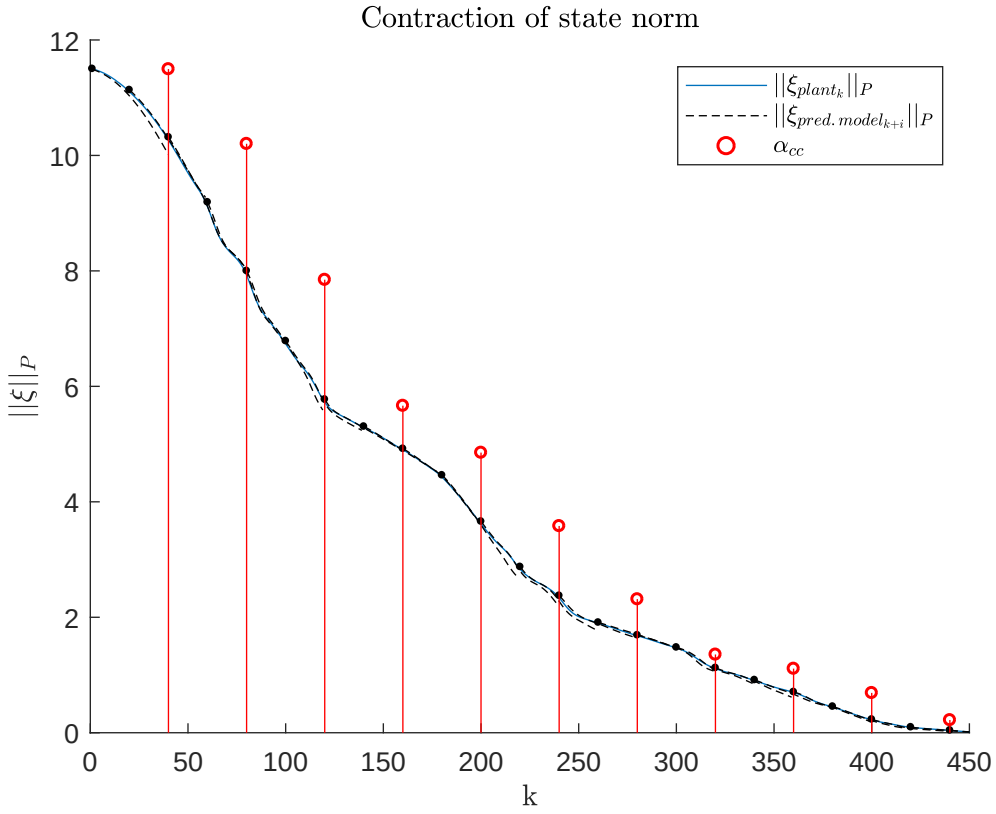


Figure 5.28: LTVMPCC contraction of state norm for Scenario 4.

In Figure 5.28 we see the contraction of the state norm during one lap. The controller has no problem to satisfy the constraint all the way down to zero. This means that the controller has decreased the state norm to satisfy the contraction constraint at each contraction step j and driven the system into the predefined equilibrium point, thus ensuring stability throughout the whole simulation.
A feasible upper bound $\hat{\alpha}$ could not be computed at every contraction step over the course of the lap due to very large nonlinearites in the model.

5.4.2 MPCC with observer

The controller in this section is the same as above but now artificial output noise is added and the states are estimated by the use of an Extended Kalman Filter. A model/plant mismatch is introduced by perturbing the Pacejka force constants. The standard values are given in 3.1. The perturbed Pacejka force constants are as follows:

Values of $\bar{\alpha}$	
Time step	$\bar{\alpha}$
$k = 40$	0.999505083703128
$k = 80$	0.993888998085892
$k = 120$	0.99
$k = 160$	0.99
$k = 200$	0.99
$k = 240$	0.99
$k = 280$	0.99
$k = 320$	0.806600365736303
$k = 360$	0.999743790777952
$k = 400$	0.99
$k = 440$	0.99

Table 5.5: Table showing values of $\bar{\alpha}$ for Scenario 4.

Pacejka formula constants	
Constant	Value
B_f	4.3
B_r	4.2
C_f	1.45
C_r	1.47
D_f	0.3
D_r	0.34

Table 5.6: Table showing values of Pacejka formula constants.

The variance for the sensors are approximated from actual data in a stationary state. However, as we want to test the controller in a more challenging way the variance for the gyro and accelerometer have been slightly increased.

The measurement values are simulated from the actual state of the vehicle by adding the corresponding disturbance to the current state. The observer uses the same variance for the observer noise as the variance for the simulated sensors.
The sample time T_s for the prediction step in the filter is $5ms$. The sample time for the sensors and and their variance is given in the following table.

Finally, the track constraints was tightened by 0.02 meters as a buffer.

Sensor sample time and variance		
sensor	sample time	variance
camera	20 ms	0.002
accelerometer	5 ms	0.0001
gyroscope	5 ms	0.005

Table 5.7: Table showing sensor sample time and their variance.

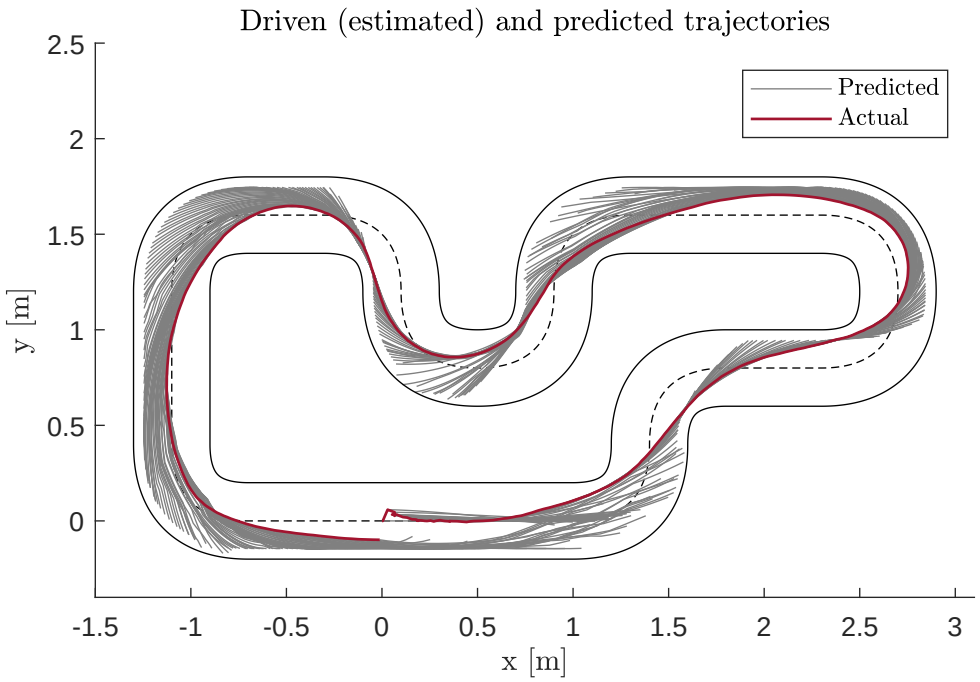


Figure 5.29: LTVMPCC with observer driven trajectories for Scenario 4.

Figure 5.29 shows the driven (estimated) and predicted trajectories for one lap. We can see that the tightening of the track constraint yields a slightly conservative driven trajectory. However in contrast to 5.25 the u-turn is navigated better. This is due to the more conservative velocity going into the u-turn.

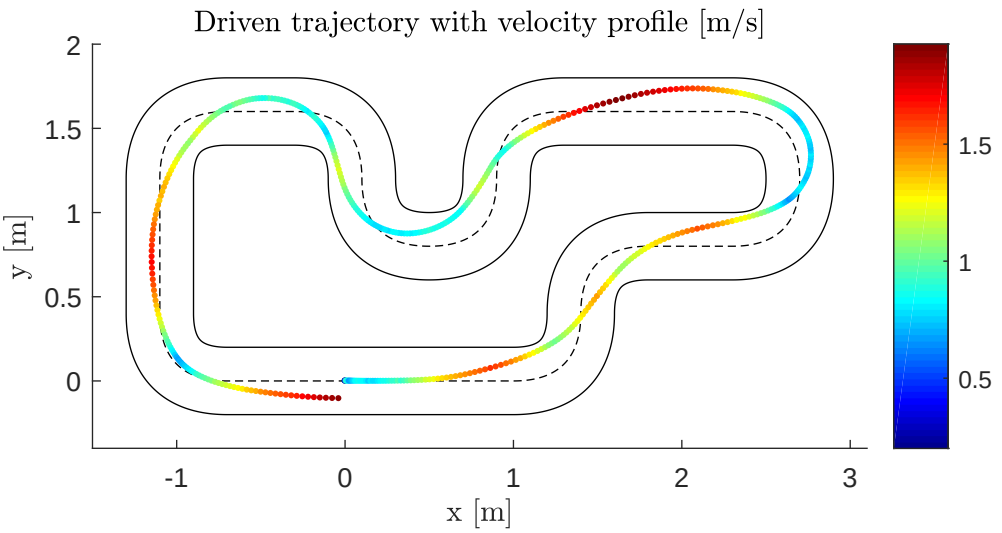
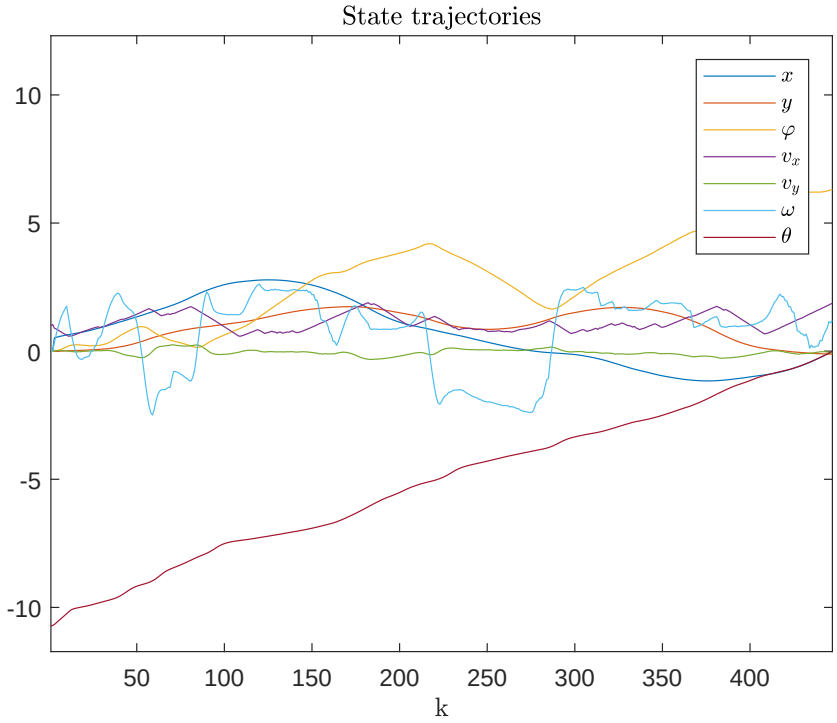
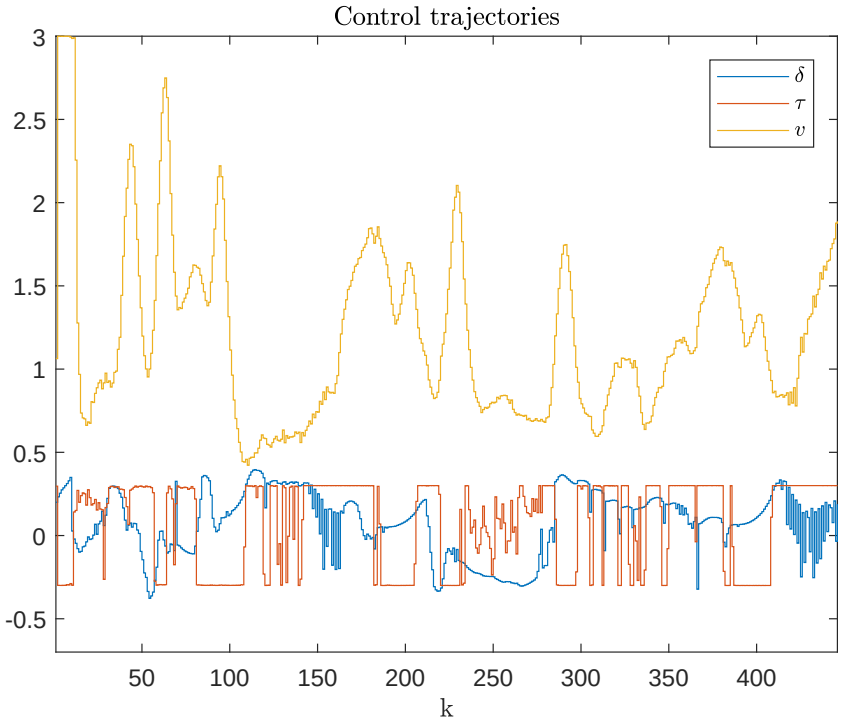


Figure 5.30: LTVMPCC with observer driven trajectory for Scenario 4.

The velocity profile in Figure 5.30 clearly shows a conservative velocity going into the u-turn. However, for the rest of the lap the controller manages to keep the velocity at a decent level despite not having full state feedback and especially more conservative track constraints. The state and control trajectories are shown in Figure 5.31.



(a) State trajectories.



(b) Control trajectories.

Figure 5.31: LTVMPCC with observer state and control trajectories for Scenario 4.

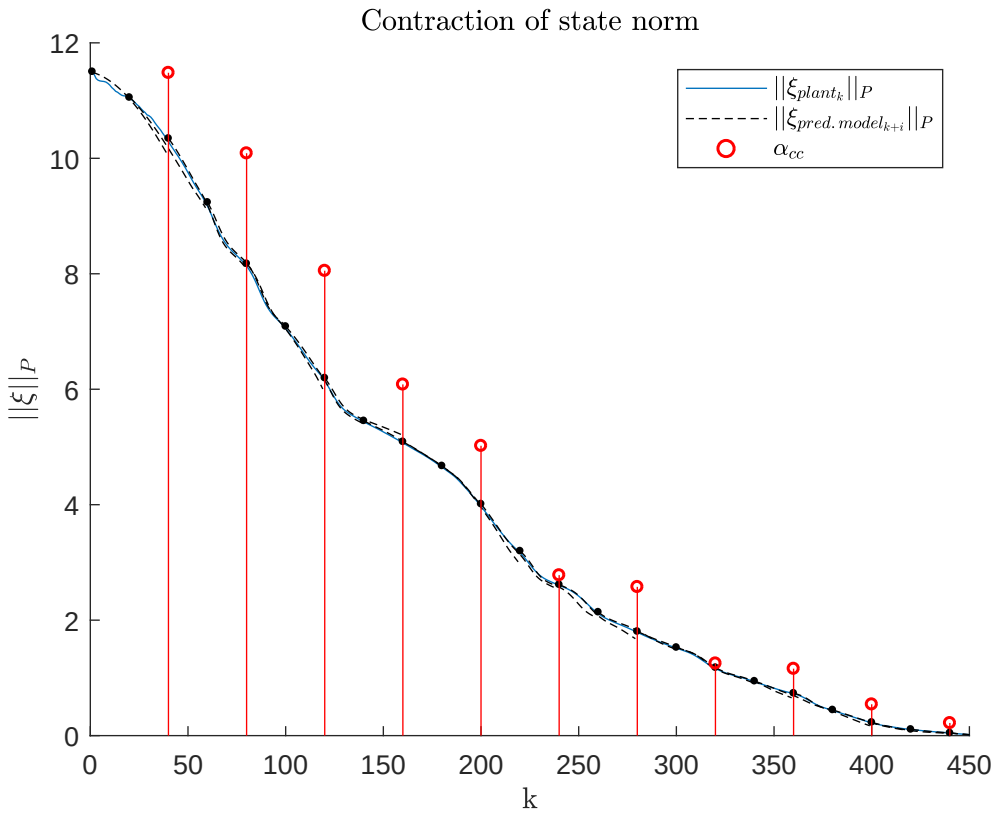


Figure 5.32: LTVMPCC with observer contraction of state norm for Scenario 4.

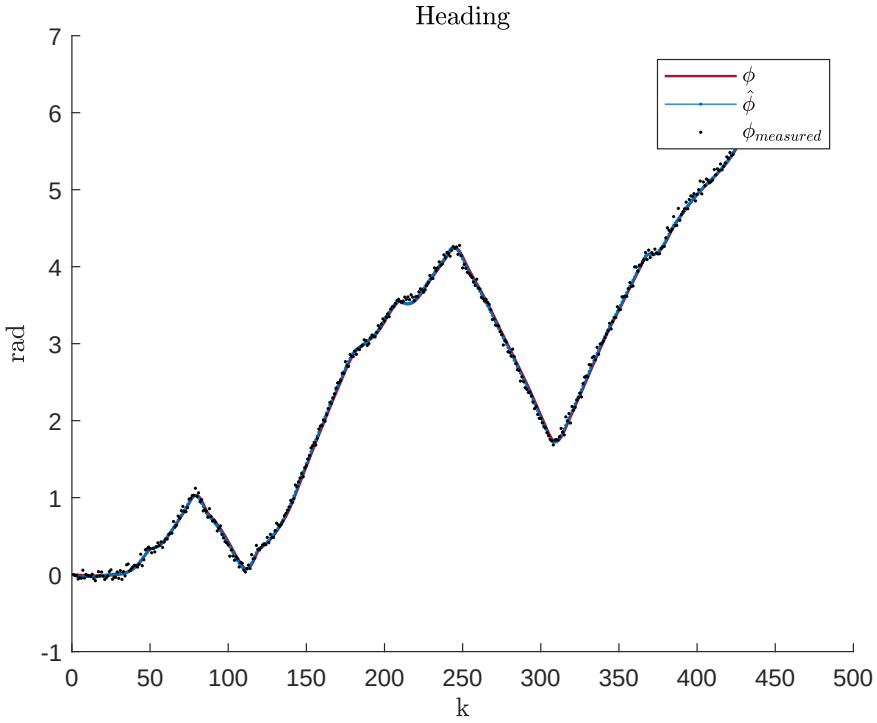
Figure 5.32 shows the contraction of the state norm during the scenario. At certain contraction steps, for example at $k = 240$ and $k = 320$, rather aggressive contraction is enforced. The controller still manages to satisfy the constraint at each contraction step.

Values of $\bar{\alpha}$	
Time step	$\bar{\alpha}$
$k = 40$	0.998282846434544
$k = 80$	0.98043673591626
$k = 120$	0.99
$k = 160$	0.99
$k = 200$	0.99
$k = 240$	0.7
$k = 280$	0.99
$k = 320$	0.7
$k = 360$	0.998673676158939
$k = 400$	0.747419733118292
$k = 440$	0.99

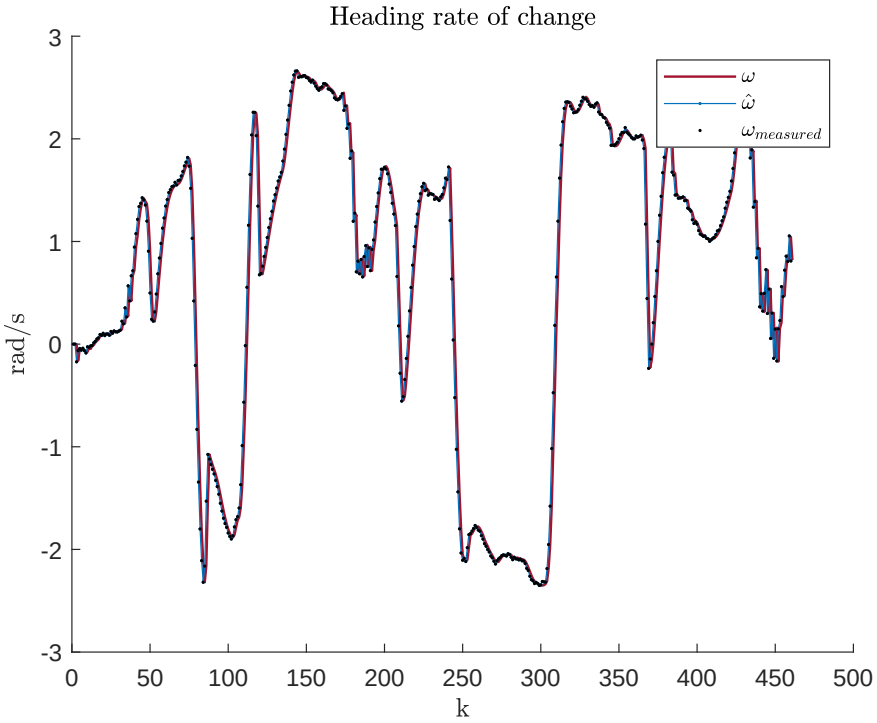
Table 5.8: Table showing values of $\bar{\alpha}$ for Scenario 4 with observer.

Table 5.8 shows the computed upper bounds on α during the course of the scenario. At time step $k = 240$ and $k = 320$ the upper bound $\bar{\alpha}$ is computed to a value of below 0.7. However, in the absence of the ability to compute a lower bound $\underline{\alpha}$ the computed value is clamped to 0.7 in order to avoid driving the system into infeasability.

Figure 5.33a and 5.33b shows the the rotation of the vehicle and it's derivative. The filtered states is almost identical to the actual states. The measurements noise is slightly higher on the angle but the filter is able to compensate for it.



(a) Heading.

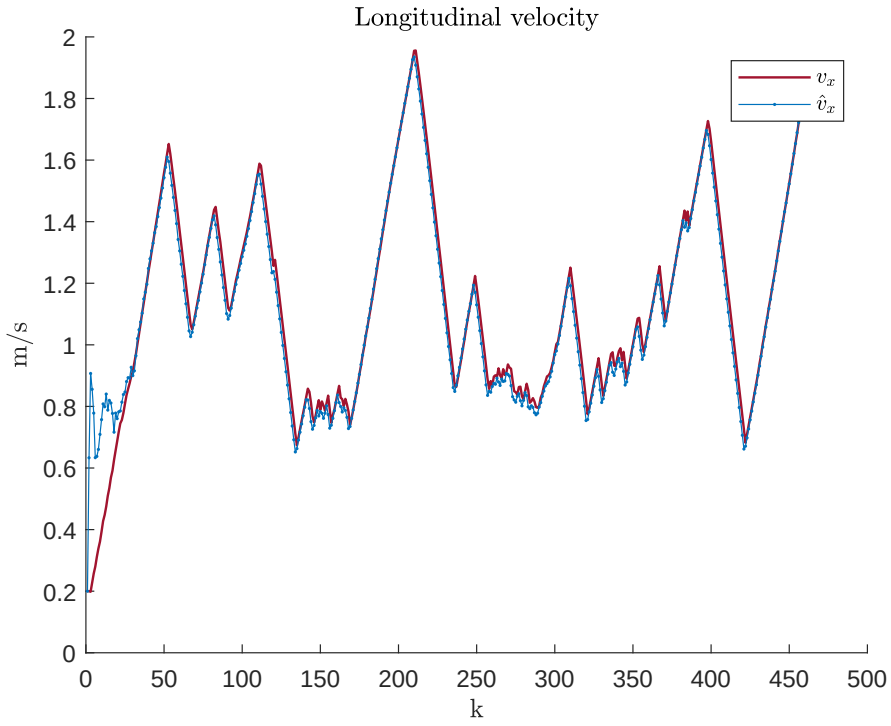


(b) Heading rate of change.

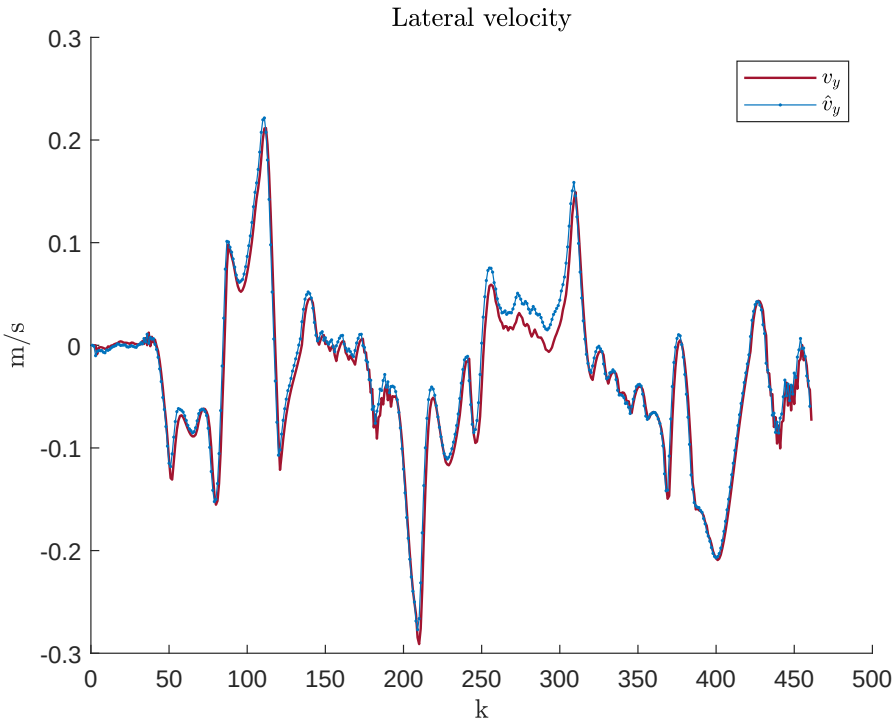
Figure 5.33: Heading and heading rate of change.

Figure 5.34a and 5.34b and shows the velocity in longitudinal and lateral direction. As the velocity states are not measured the filter has to estimate the state from current states and the relation given in the state estimation equation. The filter is rather slow to react on the sudden movement of the vehicle in the beginning of the simulation. This is mainly due to the initial value of the covariance matrix in the filter, once it has converged to a steady state the filtered becomes more accurate.

The acceleration profile in longitudinal and lateral direction is shown in Figure 5.35a and 5.35b. The figures shows the actual, measured and filtered acceleration values. The acceleration states estimated in the filter is fairly accurate and it is quick to react to the sudden changes.

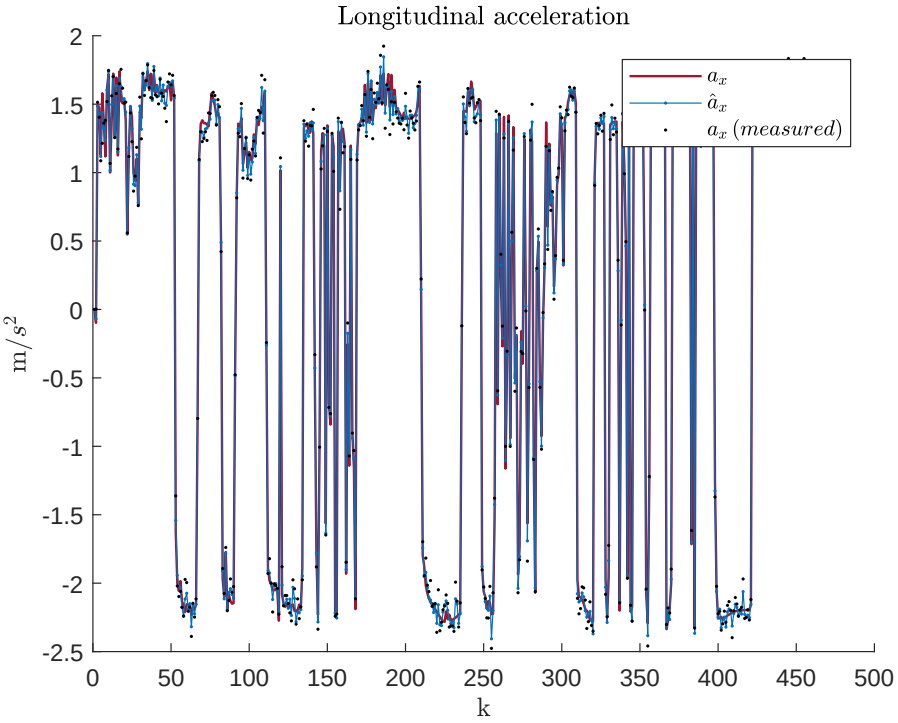


(a) Longitudinal velocity.

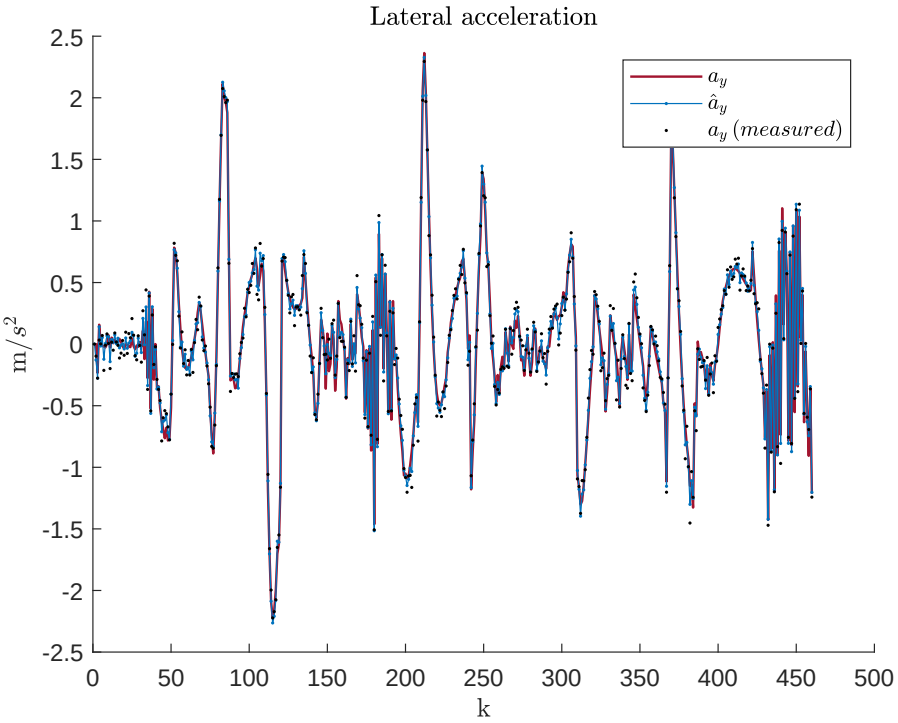


(b) Lateral velocity.

Figure 5.34: Longitudinal and lateral velocity.



(a) Longitudinal acceleration.



(b) Lateral acceleration.

Figure 5.35: Longitudinal and lateral acceleration.

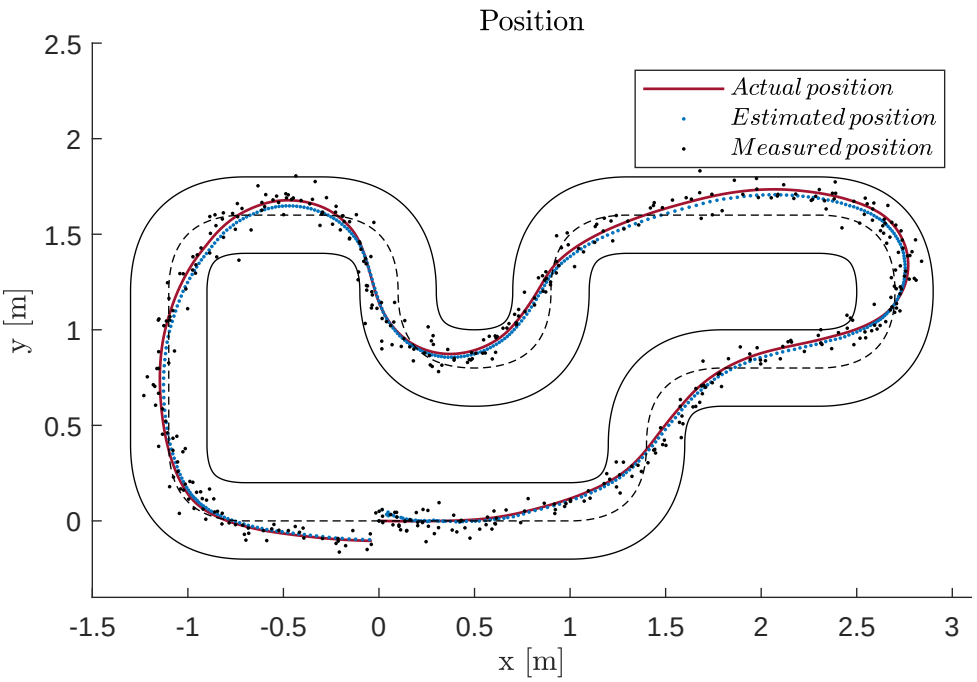


Figure 5.36: Position.

In Figure 5.36 trajectory of the vehicle is shown as well as the estimated and measured position. The position measurements are quite inaccurate but the actual and estimated trajectories are quite similar.

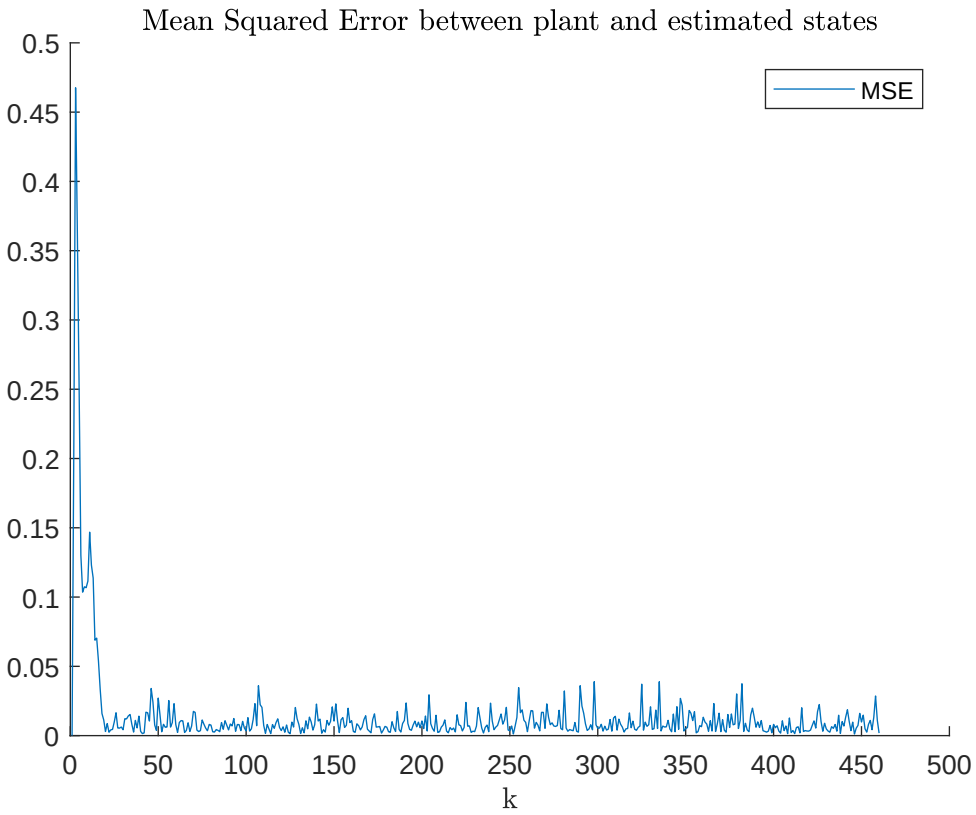


Figure 5.37: Mean squared error.

In Figure 5.37 the mean squared error between the plant and the observer states are shown. Initially the error is fairly high but decreases over the first samples and settles between 0.05 and 0. The high error in the beginning of the simulation is due to the incorrect initial guess of the covariance matrix once it has converge to a steady state the error is reduced.

5.4.3 Comparison

In Figure 5.39 the actual driven trajectories are presented for both controllers. The controller without noise manages to keep a better racing line over the track, with an exception for the u-turn. The more conservative controller with noise has an advantage here in that the car enters the u-turn section at a lower velocity and can thus manoeuvre around the u-turn without being pushed out wide. This can be seen more clearly in Figure 5.38.

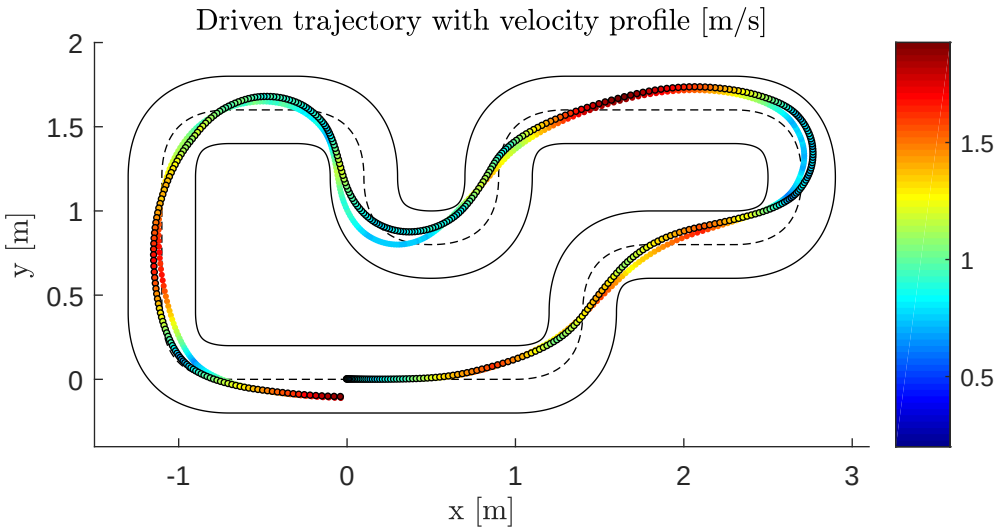


Figure 5.38: LTVMPCC and LTVMPCC with observer driven trajectories for Scenario 4.

The velocity profile with black marker edges are from the LTVMPCC with observer and the velocity profile with no marker edges are from the LTVMPCC.

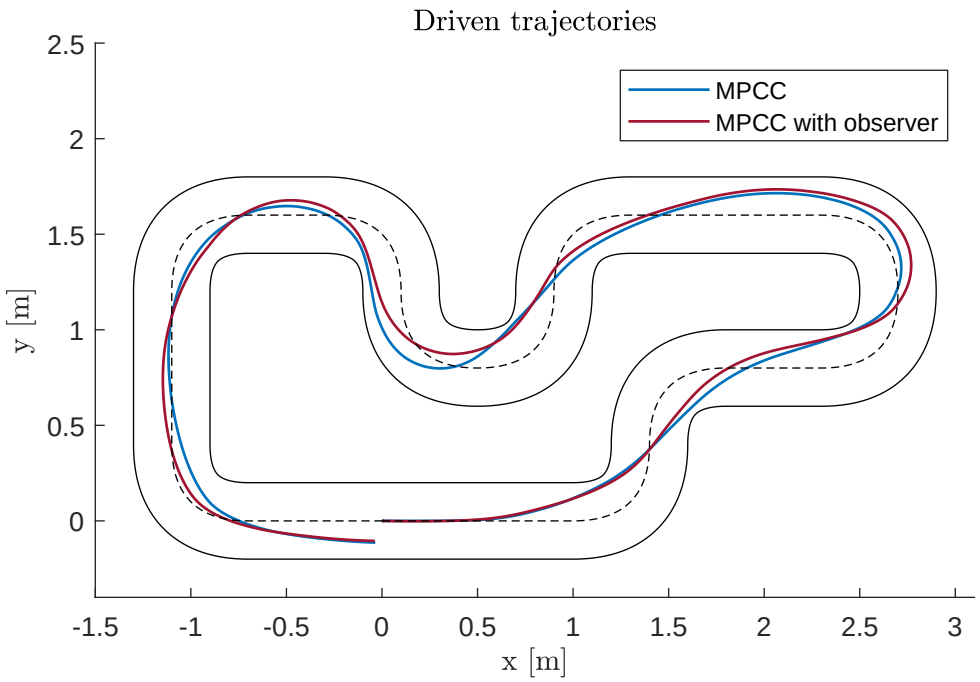


Figure 5.39: LTVMPCC and LTVMPCC with observer driven trajectories for Scenario 4.

The controller without noise finished the lap in $t_f = 9.16$ while the controller with noise finished the lap in $t_f = 9.18$. This shows that performance deterioration is minuscule when adding output noise and using estimated state feedback.

6
Conclusion

An investigation into how to use optimal control techniques, including closed-loop stability, for autonomous high-speed driving was carried out.
This thesis shows a successful implementation of a Linear Time Varying Model Predictive Contouring Controller along with an obstacle avoidance path planner. State estimation was implemented using an Extended Kalman Filter. Stabilization of the closed loop was achieved by the use of Lyapunov stability theory via a so-called contraction constraint. A unique approach for implementation of the contraction parameter computation for a linear time varying system was presented. The controller was evaluated in four different scenarios, with and without obstacle, and showed satisfactory results when compared to a time-optimal trajectory. Due to hardware restrictions the controller was regretfully not implementable with desired update rate on an embedded system.

6.1 Suggestions for further research

Finally, in this section we list some suggestions of improvements which can be made and suggestions for further research which could be of use.

- More computing power would allow for use of a longer horizon in the controller which would vastly improve driving trajectories. More computing power would also give room for use of a more advanced path planner.

- Research if the use of Nonlinear MPC is feasible with regards to solver times. This would most likely improve driving trajectories as linearization errors would no longer be present.

- Further research into different solver should be made, as CPLEX might encounter numerical difficulties when solving QCQP problems [12].

- A successful real-time hardware implementation would show how the controller would perform in the real world. Additions such as online estimation of the Pacejka parameters could be made.

Bibliography

[1] Lam, D (2012). *A Model Predictive Approach to Optimal Path-following and Contouring Control*. Ph.D. dissertation, University of Melbourne, Australia.

[2] de Oliveira, S (1996). *Model predictive control (MPC) for constrained nonlinear systems*. Ph.D. dissertation, California Institute of Technology, United States of America.

[3] A. Liniger, A. Domahidi and M. Morari (2014). *Optimization-Based Autonomous Racing of 1:43 Scale RC Cars*.

[4] L. Wenhao, J. Peng, W. Liu, J. Wang and W. Yu (2013). *A Unified Optimization Method for Real-Time Trajectory Generation of Mobile Robots with Kinodynamic Constraints in Dynamic Environment*.

[5] L. Juan, E. Jerez, K. Constantinides and G. Constantinides (2011). *A Condensed and Sparse QP Formulation for Predictive Control*.

[6] R. Jazar (2017). *Vehicle Dynamics - 3rd edition*.

[7] R.T. Uil (2007). *Tyre models for steady-state vehicle handling analysis*.

[8] P.Falcone, F. Borelli, J. Asgari, H.E. Tseng and D. Hrovat (2007). *Predictive Active Steering Control for Autonomous Vehicle Systems*.

[9] J. Gerdes and C.E Beal (2013). *Model Predictive Control for Vehicle Stabilization at the Limits of Handling*.

[10] F. Leutwiler (2016). *Nonlinear MPC for miniature RC Race Cars*.

[11] J. Yang, Z. Qu, J. Wang and K. Conrad (2010). *Comparison of Optimal Solutions to Real-Time Path Planning for a Mobile Vehicle*.

[12] IBM. *Numeric difficulties and quadratic constraints*. Available: `https://www.ibm.com/support/knowledgecenter/en/SS9UKU_12.4.0/com.ibm.cplex.zos.help/UsrMan/topics/cont_optim/qcp/16_numeric_difficulty.html`, Accessed on: 2019-03-20

A

Time varying Hessian entries

$$\circ_1 = 2q_l(\cos\phi(\hat{\theta}^*_{k+i|k}))^2 + 2q_c(\sin\phi(\hat{\theta}^*_{k+i|k}))^2$$

$$\circ_2 = 2\cos\phi(\hat{\theta}^*_{k+i|k})q_l\sin\phi(\hat{\theta}^*_{k+i|k}) - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_c\sin\phi(\hat{\theta}^*_{k+i|k})$$

$$\begin{aligned}\circ_3 =& 2q_c\sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))- \\ & - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_l(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla x_d(\hat{\theta}^*_{k+i|k}) + \nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\circ_4 = 2(\cos\phi(\hat{\theta}^*_{k+i|k})q_l\sin\phi(\hat{\theta}^*_{k+i|k}) - 2(\cos\phi(\hat{\theta}^*_{k+i|k})q_c\sin\phi(\hat{\theta}^*_{k+i|k})$$

$$\circ_5 = 2q_c(\cos\phi(\hat{\theta}^*_{k+i|k}))^2 + 2q_l(\sin\phi(\hat{\theta}^*_{k+i|k}))^2$$

$$\begin{aligned}\circ_6 =& -2q_l\sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla x_d(\hat{\theta}^*_{k+i|k}) + \nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))- \\ & - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_c(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\begin{aligned}\circ_7 =& 2q_c\sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))- \\ & - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_l(\cos\phi(\hat{\theta}^*_{k+i|k})x_d(\hat{\theta}^*_{k+i|k}) + \nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\begin{aligned}\circ_8 =& -2q_l\sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla x_d(\hat{\theta}^*_{k+i|k}) + \nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))- \\ & - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_c(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\begin{aligned}\circ_9 =& 2q_c(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))^2+ \\ & 2q_l(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla x_d(\hat{\theta}^*_{k+i|k}) + \nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))^2\end{aligned}$$

B

Time varying Gradient entries

$$\begin{aligned}\diamond_1 =&2q_c \sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}) - \sin\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k})-\\ &- \nabla x_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k})) - 2\cos\phi(\hat{\theta}^*_{k+i|k})q_l(\cos\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k})+\\ &\sin\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\begin{aligned}\diamond_2 =&-2\cos\phi(\hat{\theta}^*_{k+i|k})q_c(\cos\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}) - \sin\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k})-\\ &- \nabla x_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k})) - 2q_l\sin\phi(\hat{\theta}^*_{k+i|k})(\cos\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k})+\\ &\sin\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}))\end{aligned}$$

$$\begin{aligned}\diamond_3 =&-2q_c(\cos\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}) - \sin\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\\ &\hat{\theta}^*_{k+i|k}))(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla y_d(\hat{\theta}^*_{k+i|k}) - \nabla x_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k})) - q_t + 2q_l(\cos\phi(\hat{\theta}^*_{k+i|k})(x_d(\hat{\theta}^*_{k+i|k})-\\ &- \nabla x_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}) + \sin\phi(\hat{\theta}^*_{k+i|k})(y_d(\hat{\theta}^*_{k+i|k}) - \nabla y_d(\hat{\theta}^*_{k+i|k})\hat{\theta}^*_{k+i|k}))(\cos\phi(\hat{\theta}^*_{k+i|k})\nabla x_d(\hat{\theta}^*_{k+i|k})+\\ &\nabla y_d(\hat{\theta}^*_{k+i|k})\sin\phi(\hat{\theta}^*_{k+i|k}))\end{aligned}$$

C

Linear Quadratic controller weighting matrices

C.1 Scenario 1

$$Q_{lq} = \begin{bmatrix} 9.853 & 0 & 0 & 0 & 0 & 0 \\ 0 & 56.959 & 0 & 0 & 0 & 0 \\ 0 & 0 & 31.397 & 0 & 0 & 0 \\ 0 & 0 & 0 & 40.045 & 0 & 0 \\ 0 & 0 & 0 & 0 & 11.910 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.676 \end{bmatrix}$$

$$R_{lq} = \begin{bmatrix} 94.623 & 0 \\ 0 & 9.578 \end{bmatrix}$$

C.2 Scenario 2

$$Q_{lq} = \begin{bmatrix} 69.391 & 0 & 0 & 0 & 0 & 0 \\ 0 & 98.849 & 0 & 0 & 0 & 0 \\ 0 & 0 & 16.164 & 0 & 0 & 0 \\ 0 & 0 & 0 & 12.177 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8.371 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2.869 \end{bmatrix}$$

$$R_{lq} = \begin{bmatrix} 58.367 & 0 \\ 0 & 98.327 \end{bmatrix}$$

C.3 Scenario 3

$$Q_{lq} = \begin{bmatrix} 51.312 & 0 & 0 & 0 & 0 & 0 \\ 0 & 98.698 & 0 & 0 & 0 & 0 \\ 0 & 0 & 40.936 & 0 & 0 & 0 \\ 0 & 0 & 0 & 16.478 & 0 & 0 \\ 0 & 0 & 0 & 0 & 6.519 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2.682 \end{bmatrix}$$

$$R_{lq} = \begin{bmatrix} 48.269 & 0 \\ 0 & 48.004 \end{bmatrix}$$