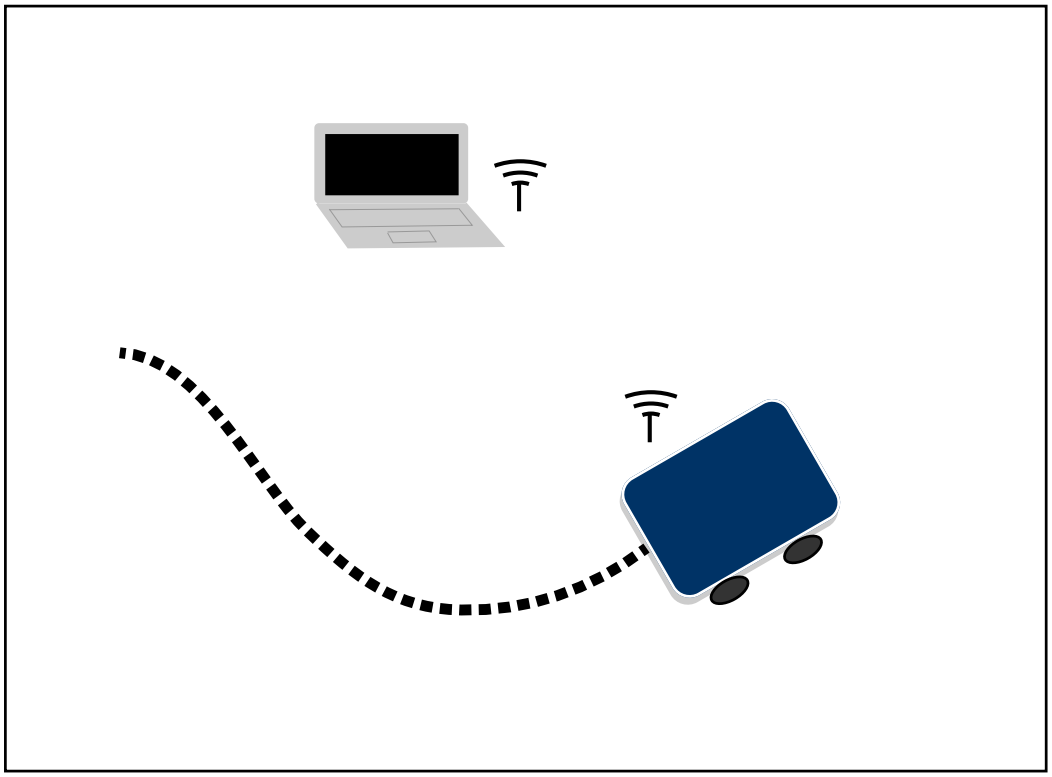
CHALMERS
UNIVERSITY OF TECHNOLOGY



Server Based Closed Loop Trajectory Control

For a Robotized Test Framework Intended for Autonomous Driving Vehicles

Master's thesis in System Control and Mechatronics

Per Ohlsson & Vanessa Olsson

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2018

Master's thesis 2018:NN

Server Based Closed Loop Trajectory Control

For a Robotized Test Framework Intended for Autonomous Driving Vehicles

PER OHLSSON VANESSA OLSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering
Chalmers University of Technology
Gothenburg, Sweden 2018

Server Based Closed Loop Trajectory Control
PER OHLSSON
VANESSA OLSSON

Supervisor: Siddhant Gupta, Volvo Cars
Examiner: Paolo Falcone, Electrical Engineering

Master's Thesis 2018:EENX30
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Visualization of a test target controlled along a trajectory by the server.

Typeset in LaTeX
Printed by [Name of printing company]
Gothenburg, Sweden 2018

Server Based Closed Loop Trajectory Control
PER OHLSSON VANESSA OLSSON
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Autonomous drive is rapidly evolving within the automotive industry. Extensive testing of all active safety functions in all stages of development is necessary before releasing the AD vehicles in traffic. To ensure productive testing one goal is to fully automate the testing of the real vehicle. To be able to test the AD functions in a car, test targets are used to trigger the different functionalities. This thesis describes the development of a setup to be used in controlling the test targets with a centralized controller steering targets along given real time updated trajectories. Mainly a central control strategy, with all computing executed centrally on a server, is researched and analyzed. The thesis handles modelling, state estimation, control and communication to create the framework. Both a MPC control strategy and a P control strategy with a feed forward prediction is investigated. Simulation results shows that both the P controller and MPC controller successfully controls the targets along the desired trajectories. Furthermore, it can be concluded that the performance of the implemented P controller is comparably better than the performance of the relatively more sophisticated MPC controller. Limitations in computing power and real time execution constraints lead to only the P controller being implemented on the real platform.

Keywords: Trajectory Control, MPC, P control, State estimation, Communication, Bicycle Model, Automated Test Systems

Acknowledgements

We would like to thank our supervisors Siddhant Gupta and Francesco Costagliola for your great support. In addition we extend a warm thanks to Goksan Isil and Junhua Chang for all help in the more technical matters.
We would also like to thank Paolo Falcone for being our examiner and supervisor and for the great advise and support throughout the thesis.

Per Ohlsson & Vanessa Olsson, Gothenburg, June 2018

Contents

List of Figures

List of Tables

List of Symbols

Symbols

X	global x position $[m]$
Y	global y position $[m]$
$\dot{x}$	longitudinal velocity $[m/s]$
$\dot{y}$	lateral velocity $[m/s]$
$\ddot{x}$	longitudinal acceleration $[m/s^2]$
$\ddot{y}$	lateral acceleration $[m/s^2]$
V	velocity $[m/s]$
a	acceleration $[m/s^2]$
t	time $[s]$
k	time step $[s]$
F	force $[N]$
I	moment of inertia $[kg \cdot m^2]$
m	mass $[kg]$
l	length of vehicle from axis to center of gravity $[m]$
L	length of vehicle from axis to axis $[m]$
C	tire cornering stiffness
δ	steering angle $[rad]$
ψ	yaw angle $[rad]$
$\dot{\psi}$	yaw rate $[rad/s]$
β	slip angle at center of gravity $[rad]$
α	slip angel at tires $[rad]$
θ	velocity angle with respect to longitudinal axis of vehicle
e_y	lateral error with respect to road $[m]$
e_ψ	yaw angle error $[rad]$
e	state vector
x	state vector
u	input vector
A	state matrix
B	input matrix
C	output matrix
D	feedthrough matrix
∂	partial derivative
Q	Kalman process noise covariance
R	Kalman measurements noise covariance
P	state error covariance
V_n	Objective function
$\mathbb{X}$	state set
$\mathbb{U}$	input set
N	prediction horizon
M	control horizon
K_p	proportional gain
K_v	understeering gradient
p	distance between reference points
f	motion model
F	motion model jacobian
q	process noise
r	measurement noise
v	innovation
S	innovation covariance
K	Kalman gain
P	state error covariance
H	measurement matrix

Subscripts

$_i$	index
$_f$	front
$_r$	rear
$_x$	longitudinal
$_y$	lateral
$_z$	vertical
$_0$	initial
$_f$	final
$_k$	time step
$_\psi$	yaw
$_{des}$	desired
$_{ref}$	reference
$_{min}$	min value
$_{max}$	max value
$_{gps}$	GPS
$_{acc}$	accelerometer
$_{mag}$	magnetometer
$_{gyr}$	gyroscope
$_{raw}$	raw measurement

Superscript

$\hat{}$	estimate
$\dot{}$	time derivative
$^+$	successor

Abbrevations

AD	Autonomous drive
AS	Active safety
BM	Bicycle model (kinematic)
CA	Constant acceleration model
CPU	Central processing unit
CT	Coordinated turn model
DBM	Dynamic bicycle model
EKF	Extended Kalman filter
GPS	Global positioning system
IP	Internet protocol
LTV	Linear time varying
MPC	Model predictive control
QP	Quadratic programming
RC	Radio controlled
TCP	Transmission control protocol
UDP	User datagram protocol
XML	Extensible markup language

1
Introduction

1.1 Background

Autonomous Driving (AD) is envisioned to be the next step to take in order to increase the safety of cars and in traffic. The Swedish government together with other organizations have a zero vision, that there will be no serious injuries or fatalities in traffic [2]. Volvo Car Corporation has their own vision 2020, that no one should be seriously injured in a new Volvo by the year 2020 [3]. To achieve these visions AD is a subject undergoing intense study in many companies and research institutes. The goal with AD is that it should outstand the human driving skills and not only be more comfortable but also excel in safety.

The new AD functions needs to be extensively tested in all stages of development. Computer simulations opens up for the possibility to run a large number of tests. Before a new function is released on the market, the function also needs to be tested in the real vehicle to ensure functionality. These type of tests is employed on a closed test track in the first stages. To test the AD functions the car needs to be stressed in order to trigger the functions under test. This can be done e.g. using different moving targets such as Radio Controlled (RC) cars, full sized cars or lowriders. A lowrider is a platform on which boxes can be loaded on in order to simulate e.g. the back of a car or a moose. In many cases this is done manually or in an open loop control. With today's technology the testing can be automated to be able to easier change the test scenarios and to make the testing more effective.

One idea is to use a Steer-by-Server algorithm that uses a simulation software, run in real time, which provides all targets with real time updated trajectories. The goal is to steer all targets along these trajectories via a server by sending the control signals to the test targets, as in Fig. 1.1. When running tests in real time, it is of high importance to have low computation time of control signal as well as few delays and information losses in the communication system. To achieve this the communication setup, using centralized or decentralized computation of control signals of the targets, will be researched. In Fig. 1.1, wireless communication 1 and 4 will result in a centralized control setup where the control signal is calculated on the server while wireless communication 2 and 3 will result in a decentralized setup where the control signal will be calculated on the targets, or some other combination/mix of this setup.

The targets will be controlled in a closed loop control algorithm to ensure trajectory

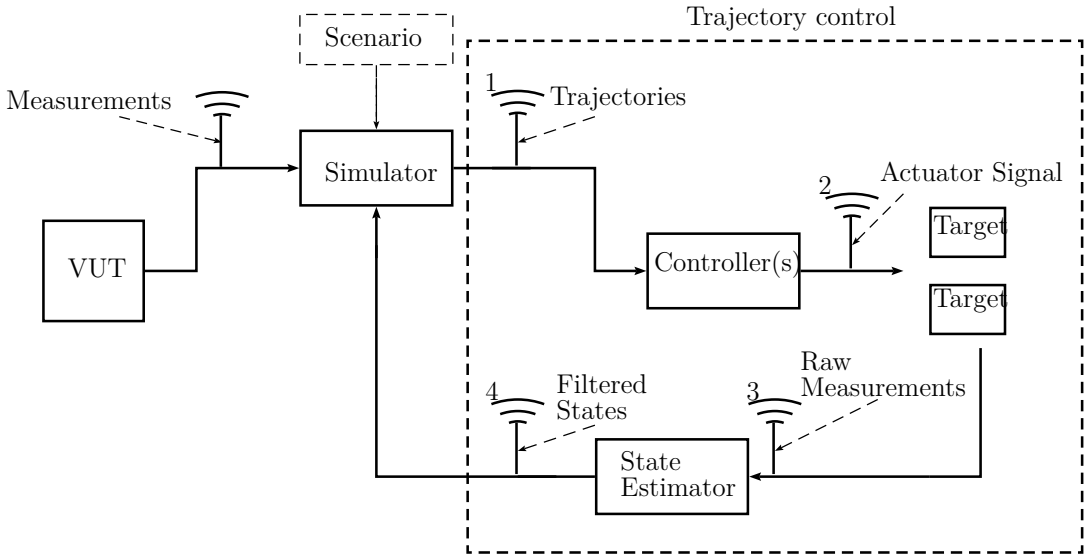


Figure 1.1: An overview over the proposed setup for the Steer-by-Server system. In this figure, the wireless communication will be either 1 or 2 for controlling the targets and 3 or 4 for feedback to the simulator.

following. This has been done in [4] and [5], in the latter, trajectory following is ensured using Kalman filter for state estimation together with a nonlinear asymptotic tracking feedback controller to steer the vehicle. For accurate control and synchronization of the different targets with respect to the simulation trajectory a state estimation of the different targets is needed. A nonlinear Kalman filter is a possible and likely candidate to use as an state observer, as in [6].

In order for a state estimator to track the RC cars and lowriders properly their motion needs to be modelled. Examples of commonly used motion models are the Constant Acceleration (CA) model, Coordinated Turn (CT) model and Bicycle model (BM). In [7] the CT model is used for a robot which is somewhat similar to a RC car. The research in [8] investigates the CT model and extensions of the CT model in more detail and concluded that the CT model in combination with a nonlinear Kalman filter achieves good tracking. In [1] vehicle dynamics is captured by a BM model.

Similarly to a control server, a fusion server to localize vehicles was proposed in [9]. The delays in the system was often caused by information packets arriving in non chronological order. The authors suggested rollback as a solution for that problem. There are different communication protocols and how suitable a protocol is depends on the application. In this case the question is where we get the best trade off regarding computation and communication. The system must be designed to be robust against communication delays and still have fast computation time. The choice can be made whether to have central or decentralized computation e.g. to send a actuator signal in form of a control signal for the acceleration and steering angle or simply send e.g. position to the target and perform all/parts of the computation on the target CPU.

As previously mentioned, there has been a lot of research in these separate areas although there is little work considering merging them together in to a complete system. The challenge is to combine the existing research and find the best setup for communication vs. control computation.

1.2 Research Questions/Objectives

The objective of this thesis work is to develop a Server based control algorithm to synchronize the trajectory of test targets with the scenario requirements. This thesis will focus on the subsystem inside the dashed line in Fig. 1.1, which will be a part of the whole Server based test system setup. A description of the subsystem focused on in this thesis can be seen in Fig. 1.2.

The algorithm should use both real time data from the test targets and trajectories, calculated by a simulator, in order to control the test targets and synchronize their positions to the trajectories. All this should be operated in real time or offline with trajectories updated simultaneously by the simulator depending on the Vehicle Under Test (VUT) behaviour and the scenario specification. All test targets should be controlled by the algorithm operating on a server. Since information will be send to and from the targets via wireless communication it is of high importance that the system is robust to delays and information losses. This setup is envisioned to ensure efficient and reproducible testing of AS and AD functions. In particular the thesis work will focus on the following aspects/questions:

- How should the test targets be modelled, in order to be controlled? On a virtual environment, suitable model fidelity of the test targets is necessary in order to create a control algorithm and steer the test targets along the trajectories.
- How does the Steer-by-Server algorithm ensure that the trajectories requested by the simulator are followed by the test targets and the said test targets are synchronized as per the scenario definition?
- What is the optimal combination of wireless communication and local computation? I.e. should the control signal be calculated centrally and be sent to the targets through wireless communication. Another alternative is to let the control signal be calculated on a local computer on the targets and be sent to the actuator, or a combination of these two? The aim is to make the control algorithm robust regarding delays and information losses.

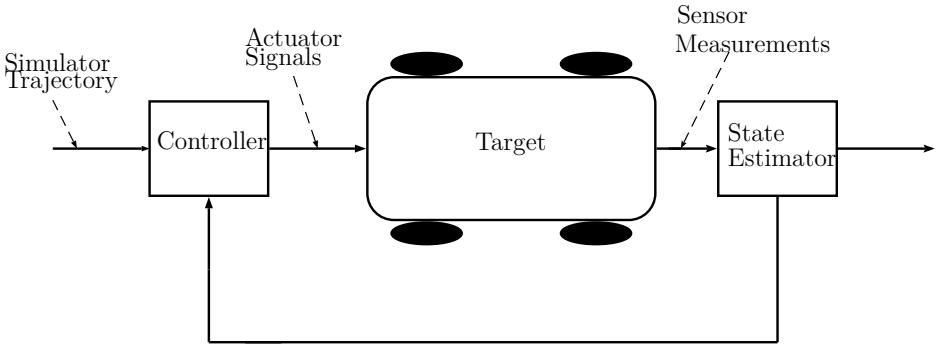


Figure 1.2: The closed loop trajectory control of the targets

1.3 Scope

The scope of the thesis is limited to trajectory control. The focus will be on implementing a versatile trajectory control algorithm which can be used to control different test actors.
The inputs to the algorithm is assumed to be feasible trajectory points, hence generation of feasible trajectories lies outside the scope of the thesis.
The aim is to use the algorithm on a real actor but this will include further work and extensive tuning and is not included in the scope.

1.4 Thesis Outline

This thesis is naturally divided in to four parts, a modelling-, a state estimation-, a control- and a communication part. In Chapter 2 the basics behind modelling is presented together with some commonly used vehicle dynamic models, the proposed model and implementation of the proposed model. A brief introduction to state estimation is presented in Chapter 3.1.1 followed by theory behind the Kalman filter and the Extended Kalman filter. In addition, a proposed state estimation setup is also presented. Two control strategies, a more complex MPC algorithm and a P control with a feed forward prediction, are presented in theory and implemented in Chapter 4. Chapter 5 touches upon communication. In this chapter basic theory regarding relevant communication is presented together with the proposed communication setup. Results from simulations and communication experiments with a RC car are presented in Chapter 6. Discussion about the work and results are given in Chapter 7. The findings are wrapped up and concluded in Chapter 8.

2
Modelling of vehicle dynamics

In order to control a vehicle along a trajectory, a model describing the dynamics is useful. There are various of kinematic models describing lateral motion, all with different complexity. If planar motion is assumed and the vehicle is not too wide, the two front wheels can be modelled compressed into one central front wheel and the rear wheels as one central rear wheel, yielding the bicycle model (BM) [1].

2.1 Kinematic Bicycle Model

The kinematic BM can be seen in Fig. 2.1, it assumes both front and rear wheel steering. The front wheel steering angle is denoted δ_f and the steering angle of the rear wheel is denoted δ_r. If the rear wheels are fixed, e.g. only front wheel steering, δ_r is set to zero.

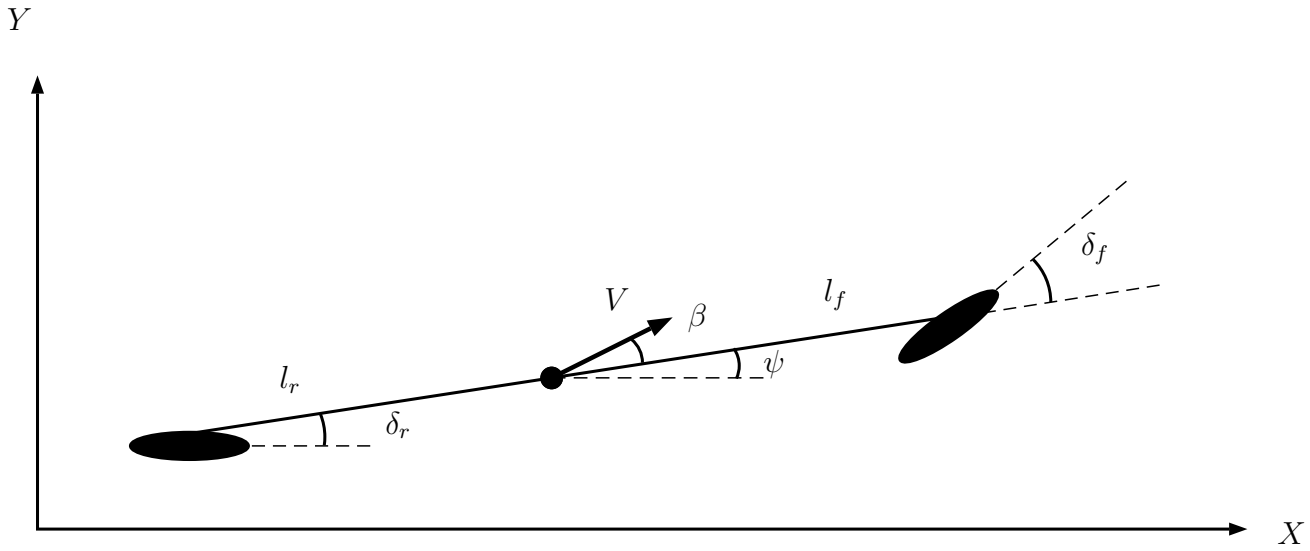


Figure 2.1: Kinematic bicycle model

The distance from the front wheel to the center of gravity is denoted by l_f and the distance from the rear wheel to the center of gravity is denoted by l_r. The slip angle of the vehicle body is denoted by β. It is defined by the angle between the velocity at the center of gravity, V, and the longitudinal axis of the vehicle. The angle ψ, the yaw or heading angle, is defined by the angle between the longitudinal axis of the vehicle and the longitudinal axis in the global coordinate system. The advantage

with the bicycle model compared to higher fidelity kinematic models is that the only parameters to identify by system identification is l_f and l_r, which are easily measured. However, there is a major assumption regarding the bicycle model, the velocity vectors at the front and rear wheel are assumed to be in the same direction as the orientation of the front and rear wheel, i.e δ_f and δ_r from the longitudinal axis of the vehicle. Another way of phrasing this, is that the slip angle at the front and rear wheel equals to zero. Hence, this assumption is only valid when the vehicle is moving at relatively low speed (speed up to 5 m/s) since the lateral tire forces are small at low velocities. When the vehicle is moving at higher velocities, the lateral tire forces can not be neglected and a model with higher fidelity is necessary for accurate modelling [1].

Equations for a kinematic bicycle model from [1]:

$$\dot{X} = Vcos(\psi + \beta) \tag{2.1}$$

$$\dot{Y} = Vsin(\psi + \beta) \tag{2.2}$$

$$\dot{V} = a \tag{2.3}$$

$$\dot{\psi} = \frac{Vcos(\beta)}{l_f + l_r}\Big(tan(\delta_f) - tan(\delta_r)\Big) \tag{2.4}$$

$$\dot{\beta} = tan^{-1}\left(\frac{l_f tan(\delta_r) + l_r tan(\delta_f)}{l_f + l_r}\right) \tag{2.5}$$

With inputs forward acceleration a, front wheel steering angle δ_f and rear wheel steering angle δ_r.

2.2 Dynamic Bicycle Model

For higher velocities the kinematic model in Eq. 2.1-2.5 does no longer apply. The dynamic bicycle model (DBM) is similar to the BM but the DBM also takes the tire forces into account. In contrast to the BM, where V is describing the velocity, in the DBM the velocities are divided into two components described as a longitudinal velocity $\dot{x}$ and a lateral velocity $\dot{y}$, both in the body frame. In the DBM the lateral and longitudinal position and the yaw angle are defined in the same way as for the BM in a global frame. Assuming front wheel steering only, the DBM is described by the following differential equations

$$\ddot{x} = \dot{\psi}\dot{y} + a_x \tag{2.6}$$

$$\ddot{y} = -\dot{\psi}\dot{x} + \frac{2}{m}(F_{c,f}cos\delta_f + F_{c,r}) \tag{2.7}$$

$$\ddot{\psi} = \frac{2}{I_z}(l_f F_{c,f} - l_r F_{c,r}) \tag{2.8}$$

$$\dot{X} = \dot{x}cos\psi - \dot{y}sin\psi \tag{2.9}$$

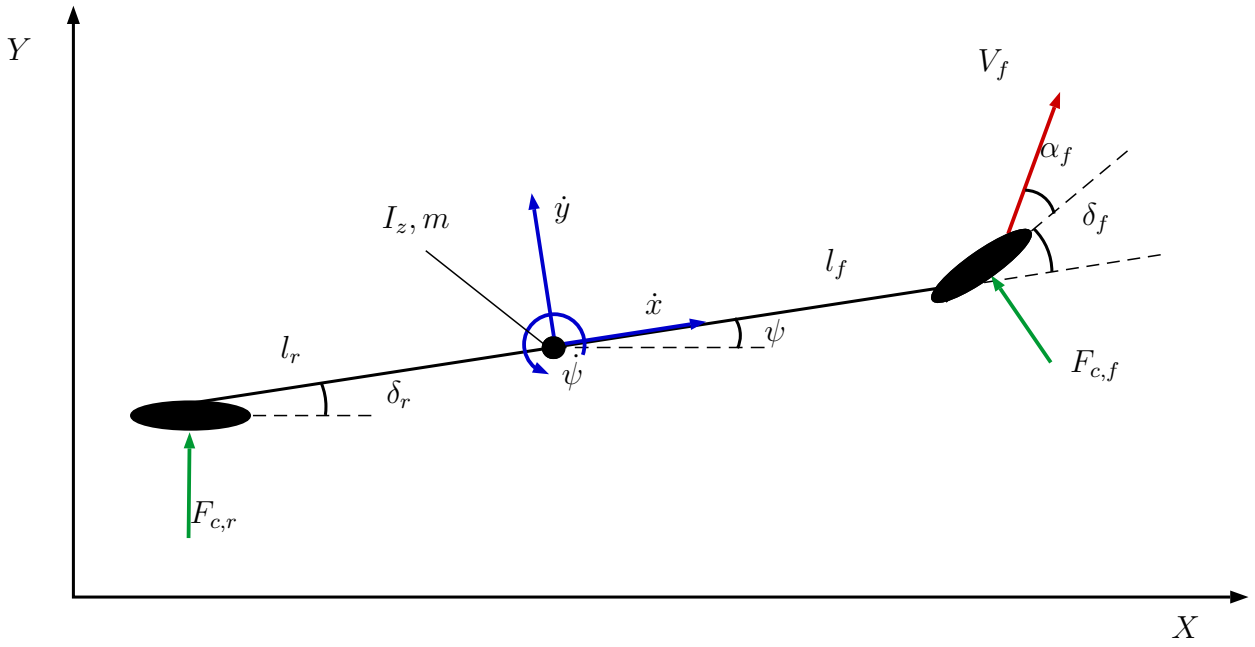


Figure 2.2: Dynamic Bicycle Model

$$\dot{Y} = \dot{x}sin\psi + \dot{y}cos\psi \tag{2.10}$$

where X is the x-position in the global coordinate frame, Y is the y-position in the global coordinate frame, $\dot{x}$ is the longitudinal velocity, $\dot{y}$ is the lateral velocity, ψ is the yaw angle and $\dot{\psi}$ is the yaw rate of the vehicle. m is the mass of the vehicle, I_z is the yaw inertia, a_x is the longitudinal acceleration in the body frame, $F_{c,f}$ is the lateral tire force on the front tire and $F_{c,r}$ is the lateral tire force on the rear tire [1], [10], [11]. Using a linear tire model, the lateral tire forces can be expressed as

$$F_{c,i} = C_i \alpha_i \tag{2.11}$$

where $i \in \{f, r\}$, C_i is the tire cornering stiffness constant and α_i is the slip angle of the wheel. The slip angle of the front wheel, α_f, can be expressed as

$$\alpha_f = \delta_f - \theta_f \tag{2.12}$$

where δ_f is the steering angle and θ_f is the angle between the front tire velocity vector and the longitudinal axis of the vehicle. The slip angle of the rear wheel can be expressed as (assuming front wheel steering only)

$$\alpha_r = -\theta_r \tag{2.13}$$

θ_r are the angle to the rear tire velocity vector with respect to the longitudinal axis. The angles θ_f and θ_r can be expressed by the following relations, [1]

$$tan(\theta_f) = \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}} \tag{2.14}$$

and

$$tan(\theta_r) = \frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}. \tag{2.15}$$

By using small angle approximations and combining Eq. 2.11-2.15 the following expressions for the front and rear tire forces are obtained

$$F_{c,f} = C_f\Big(\delta_f - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}\Big) \tag{2.16}$$

$$F_{c,r} = C_r\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}. \tag{2.17}$$

Combining Eq. 2.7, 2.8, 2.16 and 2.17 the following expressions of $\ddot{y}$ and $\ddot{\psi}$ can be derived yielding in the DBM.

$$\ddot{y} = -\dot{\psi}\dot{x} + \frac{2}{m}\left(C_f\Big(\delta_f - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}\Big)cos\delta_f - C_r\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}\right) \tag{2.18}$$

$$\ddot{\psi} = \frac{2}{I_z}\left(l_fC_f\Big(\delta_f - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}\Big) + l_rC_r\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}\right). \tag{2.19}$$

In this model the parameters l_f, l_r, m, C_f, C_r and I_z needs to be identified. The three first parameters mentioned are easily measured while the last three are a bit harder to identify and is with advantage identified with some type of system identification.

2.3 System Identification

System Identification is the process of using observed data and identifying a mathematical model of the system, this is done by observing input and output signals and describing this relation with difference or differential equations. System Identification can be applied to systems where no prior information about parameters or system properties are known, Black box modelling, and where some parameters and properties are known, Gray box modelling. Black box modelling uses models like ARX, ARMAX but also Artificial Neural Networks (ANN) etc. [12]. Gray box modelling is often used together with physical or mathematical models to estimate some unknown parameters.

One of the simplest models is the Linear difference equation model

$$y(t) + a_1y(t-1) + ... + a_ny(t-n) = b_1u(t-1) + ... + b_mu(t-m). \tag{2.20}$$

Viewing Eq. 2.20 in terms of the previous input and outputs signals determining the next output signal it can be rewritten on the following form

$$y(t) = -a_1y(t-1) - ... - a_ny(t-n) + b_1u(t-1) + ... + b_mu(t-m). \tag{2.21}$$

For simplification purposes the signals and parameters can be written in vector format. The vector containing the parameters to be estimated is written as

$$\theta = \begin{bmatrix} a_1, ..., a_n & b_1, ..., b_m \end{bmatrix}^T \tag{2.22}$$

and the vector of the previous input and output signals becomes

$$\varphi(t) = \begin{bmatrix} -y(t-1)... - y(t-n) & u(t-1)...u(t-m) \end{bmatrix}^T. \tag{2.23}$$

The output signal at time, t, rewriting Eq. 2.21 with Eq. 2.22 and 2.23 can be written as

$$y(t) = \theta^T \varphi(t). \tag{2.24}$$

2.3.1 Least Squares

In the case that there is no knowledge about the parameters in θ, but recorded inputs and outputs for time $1 < t < N$, then the recorded data can be denoted

$$Z^N = \{u(1), y(1), ..., u(N), y(N)\}. \tag{2.25}$$

The Least square method can be applied by choosing θ such that minimizing the squared error between the calculated values and the recorded values.

$$V_N(\theta) = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \hat{y}(t|\theta))^2 = \frac{1}{N}\sum_{t=1}^{N}(y(t) - \varphi^T(t)\theta)^2 \tag{2.26}$$

$$\theta_N^* = arg\min_{\theta} V_n(\theta, Z^N) \tag{2.27}$$

Differentiating Eq. 2.26 and setting equal to zero to find the minimum.

$$0 = \frac{d}{d\theta}V_N(\theta, Z^N) = \frac{2}{N}\sum_{t=1}^{N}\varphi(t)(y(t) - \varphi^T(t)\theta) \tag{2.28}$$

Rearranging Eq. 2.28 gives

$$\sum_{t=1}^{N}\varphi(t)y(t) = \sum_{t=1}^{N}\varphi(t)\varphi^T(t)\theta. \tag{2.29}$$

The estimated value θ_N^* minimizing Eq. 2.27 thus becomes

$$\theta_N^* = \left[\sum_{t=1}^{N}\varphi(t)\varphi^T(t)\right]^{-1}\sum_{t=1}^{N}\varphi(t)y(t). \tag{2.30}$$

2.3.2 System Identification Implementation

In the dynamic bicycle model Eq. 2.6-2.10 there is a set of parameters, which varies depending on the vehicle. The parameters m, l_f, l_r, I_z, C_f and C_r are the parameters which can vary to fit vehicles with different characteristics. The mass, m, and the lengths, l_f and l_r, are easily measured but the yaw inertia, I_z, of the test targets and the cornering stiffness constants,C_f and C_r, of the tires is hard to measure. A useful tool to identify the remaining parameters is system identification.

Collection of data

To estimate the parameters in Eq. 2.18 and Eq. 2.19 field data was collected. The different data sets collected was chosen to excite the system as much as possible. The field experiments was conducted with applying a constant speed to the RC car and different sinusoidal steering signals with different frequencies and amplitudes.

Parameter Identification

The parameters to identify is solely found in the DMB Eq. 2.18 and 2.19. The equations are linear in terms of the parameters to identify and can hence be identified using the linear least square method in two steps. The first step is to use Eq. 2.31 and measurements to identify C_f and C_r

$$\ddot{y} = -\dot{\psi}\dot{x} + \frac{2}{m}\left(C_f\big(\delta_f - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}\big)cos\delta_f - C_r\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}\right) \tag{2.31}$$

Rewriting Eq. 2.31 in terms of the parameters to identify, C_f and C_r, gives

$$\underbrace{\frac{m(\ddot{y} + \dot{\psi}\dot{x})}{2}}_{y_N} = \underbrace{\begin{bmatrix} C_f & C_r \end{bmatrix}}_{\theta_N^T}\underbrace{\begin{bmatrix} \big(\delta_f - \frac{\dot{y}+l_f\dot{\psi}}{\dot{x}}\big)cos\delta_f \\ -\frac{\dot{y}-l_r\dot{\psi}}{\dot{x}} \end{bmatrix}}_{\varphi_N} \tag{2.32}$$

where subscript N emphasizes that there are N samples. Using Eq. 2.32 the least square method can be used and identifying C_f and C_r. Once C_f and C_r are identified the same procedure can be repeated for Eq. 2.33 to identify I_z

$$\ddot{\psi} = \frac{2}{I_z}\left(l_f C_f\big(\delta_f - \frac{\dot{y} + l_f\dot{\psi}}{\dot{x}}\big) + l_r C_r\frac{\dot{y} - l_r\dot{\psi}}{\dot{x}}\right). \tag{2.33}$$

Rewriting 2.33 in terms of I_z gives

$$\underbrace{\frac{\ddot{\psi}}{2}}_{y_N} = \underbrace{\begin{bmatrix} \frac{1}{I_z} \end{bmatrix}}_{\theta_N^T}\underbrace{\begin{bmatrix} l_f C_f\big(\delta_f - \frac{\dot{y}+l_f\dot{\psi}}{\dot{x}}\big) + l_r C_r\frac{\dot{y}-l_r\dot{\psi}}{\dot{x}} \end{bmatrix}}_{\varphi_N} \tag{2.34}$$

I_z can then be estimated using least squares. The numerical values of the identified parameters can be found in Table 2.1.

Table 2.1: Numerical parameters of the RC car.

Parameter	Value	Unit
m	12	kg
l_f	0.23	m
l_r	0.23	m
I_z	1.7301	$kg \cdot m^2$
C_f	2.9674	
C_r	8.5430	

Model evaluation

In Fig. 2.3-2.6, how well the identified model corresponds to measurements from the RC car is presented. The measurements used in the evaluation are longitudinal velocity integrated from an accelerometer, lateral acceleration from an accelerometer, yaw rate from a gyroscope and position measurements from a GPS. It can be observed in Fig 2.3 and 2.5 that there are small offset on the longitudinal velocity as well as on the yaw rate. In Fig. 2.4 it can be seen that the identified model fits the measurements quite well even though the accelerometer measurements are noisy. Furthermore, it can be seen that the lateral acceleration of the model is more similar to to a sinusoidal than the measurements. In Fig. 2.6 it can be observed that the RC car drifts to the left, which consist with the fact that the measured yaw rate is not symmetric around zero, see Fig. 2.5.

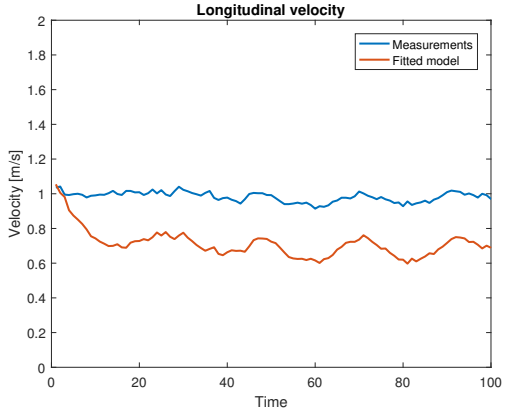


Figure 2.3: Comparison of longitudinal velocity between experimental data and open loop simulation with identified parameters.

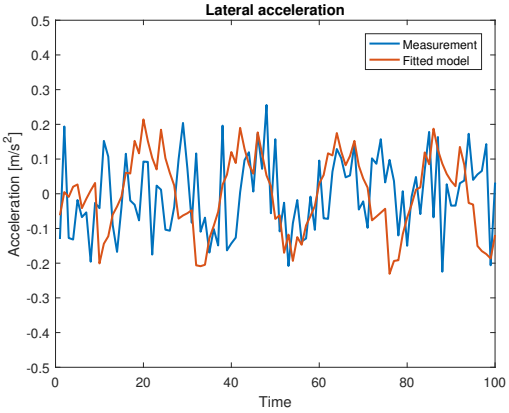


Figure 2.4: Comparison of lateral acceleration between experimental data and open loop simulation with identified parameters.

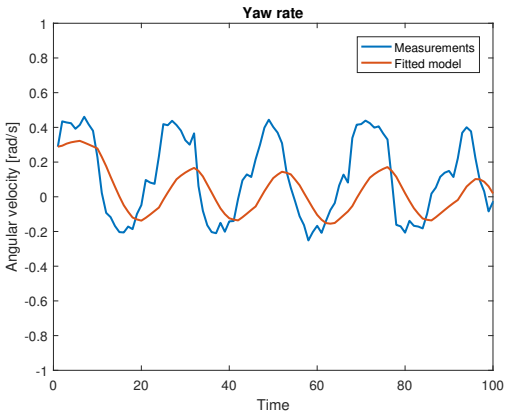


Figure 2.5: Comparison of yaw rate between experimental data and open loop simulation with identified parameters.

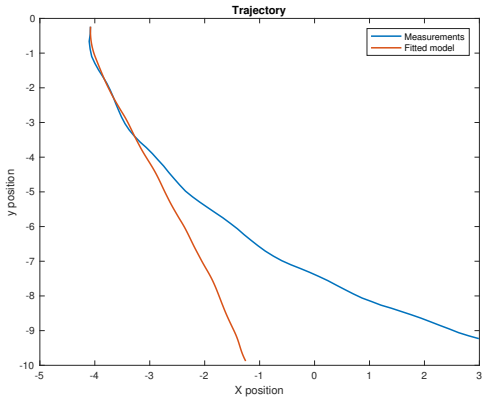


Figure 2.6: Comparison of trajectory between experimental data and open loop simulation with identified parameters.

The measurements in Fig. 2.3-2.6 were collected from an experiment where the RC car was driven with a constant velocity of 1 m/s and a sinusiodal steering with frequency 3 Hz. In Fig. 2.7 the commanded and actual steering from the experiment can be seen.

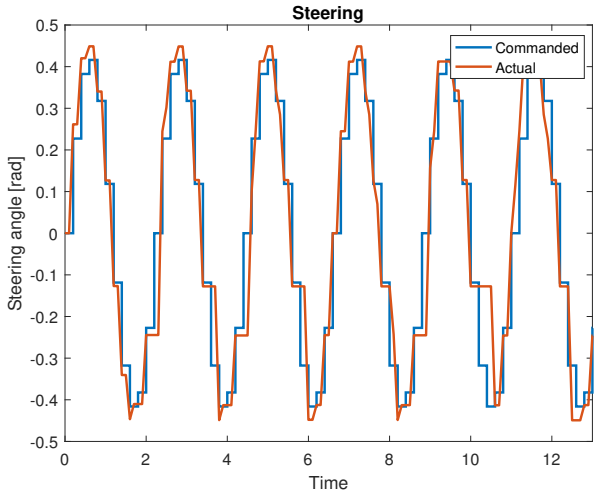


Figure 2.7: Commanded and actual steering angle.

The numerical parameters in Table 2.2 are used to simulate larger test targets.

Table 2.2: Numerical parameters of a real sized car from [1].

Parameter	Value	Unit
m	1573	kg
l_f	1.1	m
l_r	1.58	m
I_z	2873	$kg \cdot m^2$
C_f	80000	
C_r	80000	

2.4 Controller Model

When controlling the vehicle the goal is to ensure trajectory following. The state variables can then with advantage be manipulated and instead be expressed in terms of deviation variables.

The DBM Eq. 2.6-2.10 is rewritten in terms of error variables as in [1]. The position state variables are instead expressed as deviation variables from the reference, in the following manner: let e_y be the distance between the center of gravity of the vehicle and the reference trajectory. Similarly for the orientation error: let e_ψ be the orientation error of the vehicle with respect to the road. A visual representation of the error variables can be seen in Fig. 2.8. Define

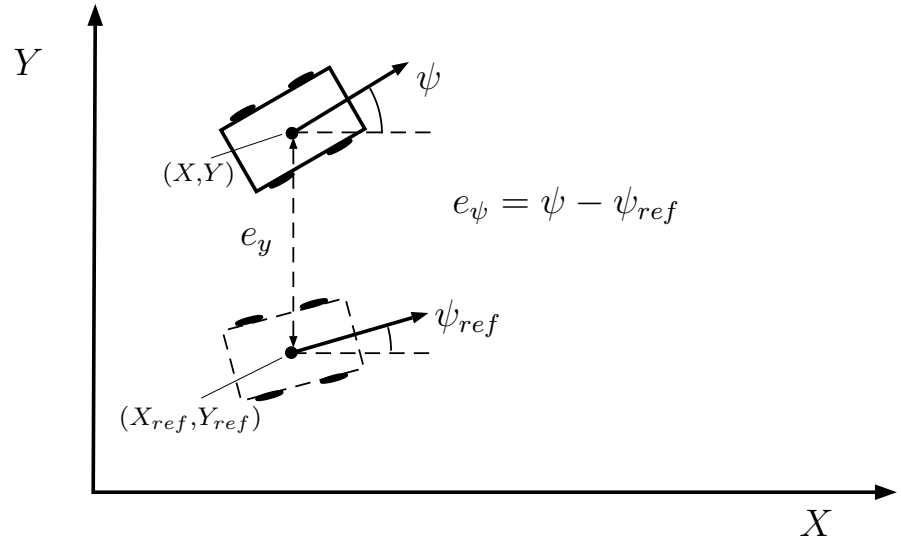


Figure 2.8: Controller model

$$\ddot{e}_y = \ddot{y} + V_x(\dot{\psi} - \dot{\psi}_{ref}) \tag{2.35}$$

$$e_\psi = \psi - \psi_{ref} \tag{2.36}$$

where V_x is the longitudinal speed of the vehicle ($V_x = \dot{x}$) [1]. The angular rate of change to reach the desired orientation can be obtained from the current reference yaw angle to the next reference yaw angle. Define the rate of change of the lateral error as

$$\dot{e}_y = \dot{y} + V_x(\psi - \psi_{ref}) \tag{2.37}$$

then Eq. 2.37 is consistent with Eq. 2.35 if the forward speed V_x is constant [1]. Substituting from 2.35 and 2.36 to 2.18 and 2.19 the state space model in terms of the new deviation models is given as

$$\begin{aligned}\ddot{e}_y = &-\frac{2C_f + 2C_r}{mV_x}\dot{e}_y + \frac{2C_f + 2C_r}{m}e_\psi + \frac{-2C_f l_f + 2C_r l_r}{mV_x}\dot{e}_\psi \\ &+ \frac{2C_f}{m}\delta_f - \frac{2C_f l_f - 2C_r l_r}{mV_x}\dot{\psi}_{ref}\end{aligned} \tag{2.38}$$

$$\begin{aligned}\ddot{e}_\psi = &-\frac{2C_f l_f - 2C_r l_r}{I_z V_x}\dot{e}_y + \frac{2C_f l_f - 2C_r l_r}{I_z}e_\psi - \frac{2C_f l_f^2 + 2C_r l_r^2}{I_z V_x}\dot{e}_\psi \\ &+ \frac{2C_f l_f}{I_z}\delta_f - \frac{2C_f l_f^2 + 2C_r l_r^2}{I_z V_x}\dot{\psi}_{ref}.\end{aligned} \tag{2.39}$$

By observing the error model 2.38-2.39 one can see that the model is linear if the the longitudinal speed V_x is constant. In the case where V_x is not constant the error model becomes a linear parameter varying system with V_x as the only time varying parameter.

The errors e_y and e_ψ are described in internal coordinates and the global position coordinates of the vehicle can be obtained by

$$X = X_{ref} - e_y sin(\psi) \tag{2.40}$$

$$Y = Y_{ref} + e_y cos(\psi) \tag{2.41}$$

where X_{ref} and Y_{ref} are the global coordinates of the reference trajectory.

3
State estimation

In any system the true state values that describes the system behavior cannot be exactly determined. However, the states of the system can be estimated by different filtering techniques using measurements and or models. This chapter addressees state estimation. Basics about the Kalman filter and the Extended Kalman filter (EKF) will be presented followed by the implementation of an EKF.

3.1 Kalman filter

The Kalman filter is a filter which uses Baysian probability theory to estimate the states of a system in an optimal way. For a discrete linear Gaussian state space model the Kalman filter is the linear minimum mean square error estimator. The discrete linear state space model of a system is given by

$$x_k = A_{k-1}x_{k-1} + q_{k-1}, \quad q_{k-1} \sim \mathcal{N}(0, Q) \tag{3.1}$$

$$y_k = H_k x_k + r_k, \quad r_k \sim \mathcal{N}(0, R) \tag{3.2}$$

where x is the state vector, A_{k-1} is the transition matrix of the dynamic system, q is Gaussian process noise, y is the measurement vector, H_k is the measurement model matrix and r_k is Gaussian measurement noise. The Kalman filter estimates states in a recursion including two steps, first a prediction followed by an update step [13].

The prediction step uses a model of the process and predicts the states according to:

$$\hat{x}_{k|k-1} = A_{k-1}\hat{x}_{k-1|k-1} \tag{3.3}$$

$$P_{k|k-1} = A_{k-1}P_{k-1|k-1}A_{k-1}^T + Q \tag{3.4}$$

where $\hat{x}_{k|k-1}$ is the predicted state at time k, $\hat{x}_{k-1|k-1}$ is the estimated state at time $k-1$, $P_{k|k-1}$ is the predicted state error covariance at time k, $P_{k-1|k-1}$ is the estimated state error covariance at time $k-1$ and Q is the process noise covariance matrix.

The update step uses a model of the available measurements to update the estimate according to:

$$v_k = y_k - H_k\hat{x}_{k|k-1} \tag{3.5}$$

$$S_k = H_k P_{k|k-1}H_k^T + R \tag{3.6}$$

$$K_k = P_{k|k-1} H_{k|k-1}^T S_k^{-1} \tag{3.7}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k v_k \tag{3.8}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T \tag{3.9}$$

where v_k is the innovation at time k, S_k is the innovation covariance at time k, R is the measurement noise covariance matrix, K_k is the Kalman gain at time k, $\hat{x}_{k|k}$ is the estimated state at time k and $P_{k|k}$ is the estimated state error covarianvce at time k [13].

3.1.1 Extended Kalman filter

As mentioned above, the Kalman filter is the linear minimum mean square error estimator. However, not all process and measurement models are linear and the ordinary Kalman filter cannot be applied for nonlinear models. If the model is nonlinear an intuitive approach is to linearize the process and measurement model around the current estimate to approximate the nonlinear models, yielding the Extended Kalman filter.

A discrete nonlinear state space representation of a system is given by the following equations

$$x_k = f(x_{k-1}, u_{k-1}) + q_{k-1}, \quad q_{k-1} \sim \mathcal{N}(0, Q) \tag{3.10}$$

$$y_k = h(x_k) + r_k, \quad r_k \sim \mathcal{N}(0, R) \tag{3.11}$$

where, x is the state vector, f is the nonlinear motion model, q is Gaussian process noise, y is the measurement vector, h is the nonlinear measurement model and r is Gaussian measurement noise. Both the process noise and the measurement noise are assumed to be Gaussian.

The prediction step:

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_{k-1}) \tag{3.12}$$

$$P_{k|k-1} = F(\hat{x}_{k-1|k-1}) P_{k-1|k-1} F(\hat{x}_{k-1|k-1})^T + Q \tag{3.13}$$

where $\hat{x}_{k|k-1}$ is the predicted state at time k, $\hat{x}_{k-1|k-1}$ is the estimated state at time $k-1$, f is the nonlinear motion model, u_{k-1} is the input signal at time $k-1$, $F(\hat{x}_{k-1|k-1}) = \frac{\partial f(x)}{\partial x}\Big|_{\hat{x}_{k-1|k-1}}$ is the Jacobian of f evaluated at $\hat{x}_{k-1|k-1}$, $P_{k|k-1}$ is the predicted state error covariance at time k, $P_{k-1|k-1}$ is the estimated state error covariance at $k-1$ and Q is the process noise covariance.

The update step:

$$S_k = H(\hat{x}_{k|k-1}) P_{k|k-1} H(\hat{x}_{k|k-1})^T + R \tag{3.14}$$

$$K_k = P_{k|k-1} H(\hat{x}_{k|k-1})^T S_k^{-1} \tag{3.15}$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\Big(y_k - h(\hat{x}_{k|k-1})\Big) \tag{3.16}$$

$$P_{k|k} = P_{k|k-1} - K_k S_k K_k^T \tag{3.17}$$

where S_k is the innovation covariance at time k, h is the nonlinear measurement model, $H(\hat{x}_{k|k-1}) = \frac{\partial h(x)}{\partial x}\Big|_{\hat{x}_{k|k-1}}$ is the Jacobian of h evaluated at $\hat{x}_{k|k-1}$, R is the measurement noise covariance, K_k is the Kalman gain at time k, $y_k - h(\hat{x}_{k|k-1})$ is the innovation at time k, $x_{k|k}$ is the estimated state at time k and $P_{k|k}$ is the estimated state error covariance at time k [13].

3.2 EKF implementation

In the framework a state estimator is necessary to ensure good control of the targets, due to noisy measurements. An EKF, as in Section 3.1.1, was implemented using the nonlinear DBM from Section 2.2 to have a good estimate of the states.

3.2.1 EKF Prediction

The prediction step in the EKF utilizes a process model of the vehicle to predict the future values of the states. The process model used in the implemented EKF is the DBM in Eq. 2.6-2.10 together with the parameters in Tab. 2.1.
The states, $\hat{x}$ are predicted using the DBM, the previous estimated states, $\hat{x}_{k-1}$ and the prevoius input u_{k-1}. To predict the covariance, P, the DBM is linearized each iteration around the previous updated estimate $\hat{x}_{k-1}$ and calculated as in Eq. 3.13.

3.2.2 EKF Update

The main idea behind the update step is to correct the predicted estimate by using information from the current sensor measurements. Often does not the measurement output we get from sensors coincide with our states. If that is the case measurement models can be used to express the measurements in terms of states. The following sensors are used when implementing the measurement model:

- Accelerometer measuring acceleration in x, y and z directions.
- Gyroscope measuring angular velocity around x, y and z axis.
- Magnetometer measuring the magnetic field in x, y and z directions.
- GPS measuring the global x and y position.

Position

The position measurements can be read directly from the GPS and simply modelled with some additive noise, denoted by r_{gps}

$$X = X_{gps} + r_{gps} \tag{3.18}$$

$$Y = Y_{gps} + r_{gps}. \tag{3.19}$$

Velocities

The velocities $\dot{x}$ and $\dot{y}$ can be calculated by numerically integrate the accelerometer measurement

$$\dot{x} = \dot{x}_0 + \sum a_x \cdot dt + r_{acc,x} \tag{3.20}$$

$$\dot{y} = \dot{y}_0 + \sum a_y \cdot dt + r_{acc,y} \tag{3.21}$$

where $\dot{x}_0$ is the initial longitudinal velocity, $\dot{y}_0$ is the initial lateral velocity, a_x is the measured longitudinal acceleration, a_y is the measured lateral acceleration, dt is the sampling time and $r_{acc,x}$ and $r_{acc,y}$ are measurement noise.

Yaw and Yaw-rate

Gyroscope measures angular velocity hence the measurement of the yaw-rate can be modelled as

$$\dot{\psi} = \dot{\psi}_{gyr} + r_{gyr} \tag{3.22}$$

where $\dot{\psi}_{gyr}$ is the gyroscope measurement and r_{gyr} is measurement noise.

The absolute orientation in the plane, the yaw angle, can be estimated using the magnetometer measurements

$$\psi = tan^{-1}\left(\frac{y_{mag}}{x_{mag}}\right) + r_{mag} \tag{3.23}$$

where x_{mag} is the measured magnetic field around the x-axis, y_{mag} is the measured magnetic field around the y-axis and r_{mag} is measurement noise.

Magnetometer Calibration

The magnetometer must be calibrated if the values are to make sense. The magnetic measurements are easily distorted by electric appliances, batteries or iron. The suggested calibration is in two steps and will give more accurate values. The two steps are correcting the measurements regarding soft iron distortion and hard iron distortion. A commonly used technique for visualizing and correcting the magnetometer measurements from soft and hard iron distortion is to slowly rotate the magnetometer around its z-axis and plot the x- and y-components of the magnetometer measurements in a 2D graph. In the case where no soft or hard iron distortions are present the measurements will form a circle with radius equal to the magnitude of the magnetic field. The circle will have its center in the origin.

Hard iron errors

Hard iron errors will only shift the center of the measurements, see Fig. 3.1.

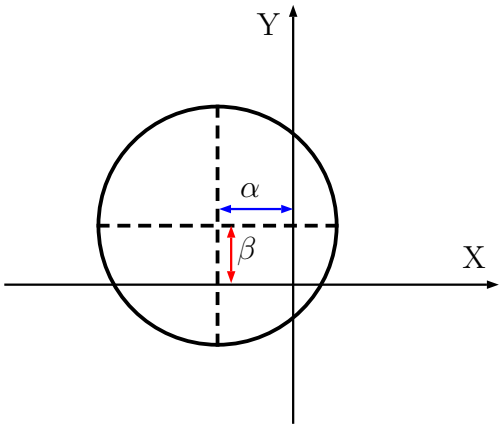


Figure 3.1: Hard iron distortion

The raw magnetometer measurements can be corrected from hard iron distortion by using

$$\alpha = \frac{x_{min} + x_{max}}{2}, \quad \beta = \frac{y_{min} + y_{max}}{2} \tag{3.24}$$

$$i_{mag} = i_{raw} - \frac{i_{min} + i_{max}}{2} \tag{3.25}$$

where $i \in \{x, y\}$, $\frac{i_{min}+i_{max}}{2}$ is the coordinate of the center of the shifted circle and subscript raw indicates the raw magnetometer measurement.

Soft iron errors

Soft iron errors will distort the measurements to an ellipse, as can be seen in Fig. 3.2. The transformation from a circle into an ellipse does not affect the center, hence

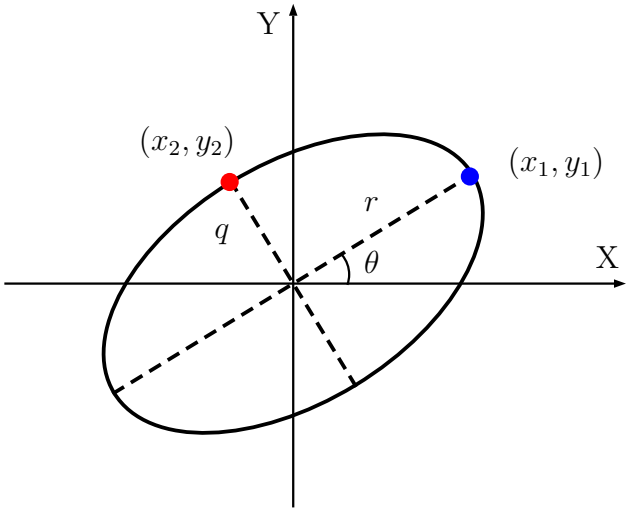


Figure 3.2: Soft iron distortion

the center of the measurements is the same as the case without soft iron distortion.

The transformation from a circle into an ellipse can be described by

$$\begin{bmatrix} x_{mag} \\ y_{mag} \\ z_{mag} \end{bmatrix} = \begin{bmatrix} C_1 & C_2 & C_3 \\ C_4 & C_5 & C_6 \\ C_7 & C_8 & C_9 \end{bmatrix} \begin{bmatrix} x_{raw} \\ y_{raw} \\ z_{raw} \end{bmatrix} \tag{3.26}$$

where, $C_3 = C_6 = C_7 = C_8 = 0$ and $C_9 = 1$ since the transformation does not affect the z-component of the measurements. The length of the major axis of the ellipse, r, is calculated by

$$r = \sqrt{x_1^2 + y_1^2} \tag{3.27}$$

where x_1 and y_1 are the coordinates on the major axis, see Fig. 3.2. The angle that the ellipse is rotated with respect to the x-axis, θ, is calculated by

$$\theta = arcsin\left(\frac{y_1}{r}\right). \tag{3.28}$$

The rotation matrix

$$R = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.29}$$

rotates the ellipse such that the major axis is aligned with the x-axis. The scaling factor between the major and minor axis, σ, is calculated by

$$\sigma = \frac{q}{r} \tag{3.30}$$

where q is the length of the minor axis and it is calculated by

$$q = \sqrt{x_2^2 + y_2^2} \tag{3.31}$$

where x_2 and y_2 are the coordinates on the minor axis, see Fig. 3.2. Only the x value is then divided by the scaling factor σ in order to transform the ellipse to a circle.

EKF Measurement Model

Since all the states in the DBM Eq. 2.6-2.10 can either be measured directly from the sensor readings or calculated from sensor readings the measurement model used in the EKF is written as

$$y_k = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}}_{H_k} x_k + \underbrace{\begin{bmatrix} r_{gps} \\ r_{gps} \\ r_{acc,x} \\ r_{acc,y} \\ r_{mag} \\ r_{gyr} \end{bmatrix}}_{r}. \tag{3.32}$$

By using pre-calculations on the sensor readings, one can observe in Eq. 3.32 that the measurement model used in the EKF is linear. This means that linearization is only necessary in the prediction step.

3.2.3 Tuning

To get a good estimate extensive tuning of the EKF is required. The tuning parameters in the EKF are the Q and R matrices. The individual values on Q and R are not important but rather the relationship between the values. The Q matrix describes the uncertainty in the model while the R matrix describes the uncertainty in the measurements. A higher value in the Q matrix makes the filter trust the measurements more and vice versa. The weight matrices Q and R, when used to estimate the states of the RC car, were tuned to

$$Q = \begin{bmatrix} 0.02 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.02 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.4 \end{bmatrix} \tag{3.33}$$

$$R = \begin{bmatrix} 0.1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.6283 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.1571 \end{bmatrix} \tag{3.34}$$

Experimental results of the EKF with the RC car and the model from 2.2 with identified parameters from Table 2.1 are shown in Fig. 3.3-3.7.

The GPS has high accuracy leading to a low corresponding value in the R matrix and it can be observed in Fig. 3.3 that the estimated trajectory is close to the measured trajectory.

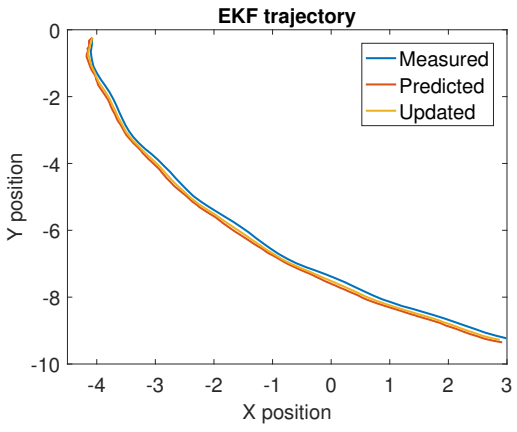


Figure 3.3: Measurements, EKF prediction and EKF update of RC car trajectory.

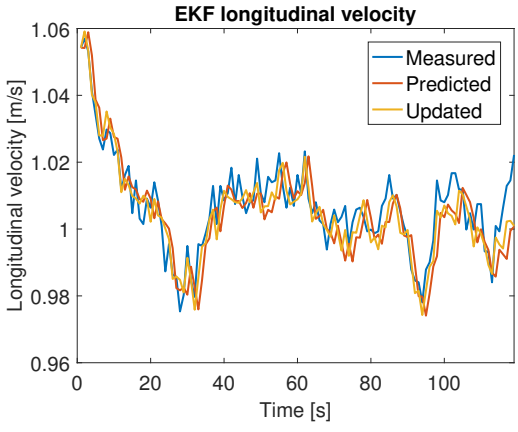


Figure 3.4: Measurements, EKF prediction and EKF update of RC car longitudinal velocity ($\dot{x}$).

In Fig. 3.4 it can be observed that the measurement of the longitudinal velocity is accurate, hence the EKF is tuned to rely more on the measurement. In Fig. 3.5 it

can be seen that the updated lateral velocity is between the predicted and measured lateral velocity most of the time. This can be explained by the fact that the EKF is tuned to rely on the model nearly as much as on the measurements.

As seen in Fig. 3.6 the magnetometer is not a very accurate sensor. The measurements are noisy leading to the choice of having a high covariance on the yaw angle in the R matrix and leading to trusting the model more.

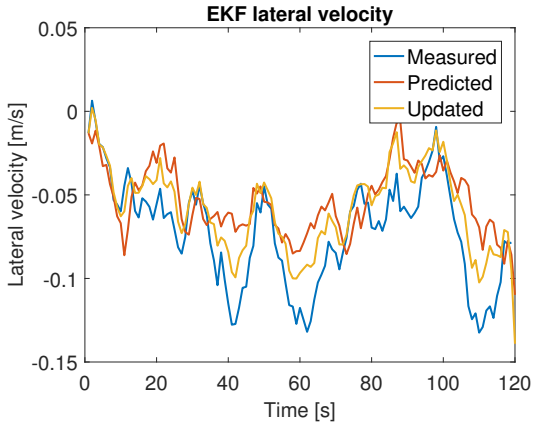


Figure 3.5: Measurements, EKF prediction and EKF update of RC car lateral velocity ($\dot{y}$).

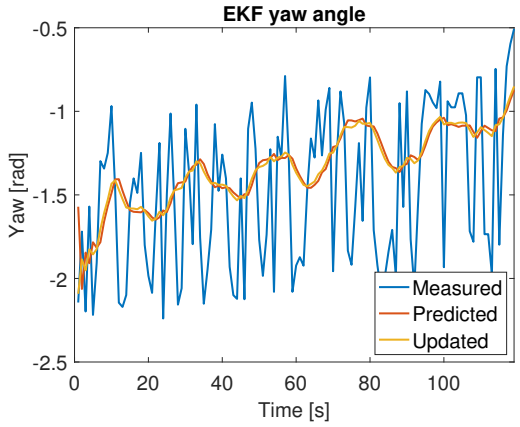


Figure 3.6: Measurements, EKF prediction and EKF update of RC car yaw angle (ψ).

The gyroscope is a sensor with high precision and the measurement can be trusted, leading to a low value on the covariance on the yaw rate measurement. The estimated yaw rate can be seen in Fig. 3.7 and it can be observed that the predicted and updated yaw rate almost overlaps the measurements.

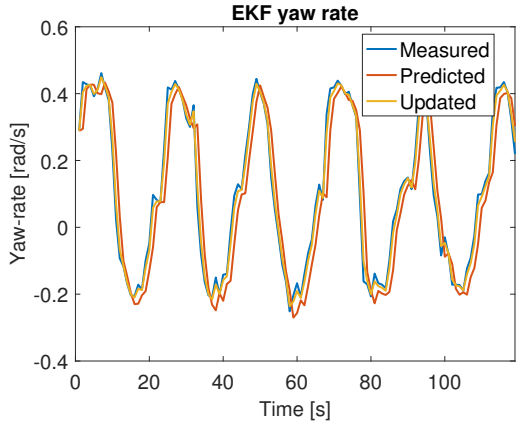


Figure 3.7: Measurements, EKF prediction and EKF update of RC car yaw rate ($\dot{\psi}$).

4

Control

To ensure the targets follows the given trajectories a good control algorithm needs to be implemented. In this chapter a MPC control strategy and a P control strategy with a feed forward prediction are investigated and implemented.

4.1 Model Predictive Control

The main idea behind the Model Predictive Control (MPC) framework is to predict the future behavior of the system, by utilizing a model of the process. In MPC the control input is obtained by solving an optimal open-loop control problem over a finite horizon. The solution to the optimal open-loop control problem is a sequence of future optimal control inputs, but only the first control input in the sequence is applied to the system. The optimal open-loop control problem is solved at each sampling instant starting from the current state and with shifted time horizon [14], [15].

4.1.1 Optimization

For a discrete time state space model the finite horizon optimal control problem can be written as

$$\min_{u(0:N-1)} \quad V_N = \sum_{i=0}^{N-1} l(x(i), u(i)) + V_f(x(N)) \tag{4.1}$$

$$s.t. \quad x(k+1) = f(x(k), u(k)), \quad x(0) = x_0 \tag{4.2}$$

$$y(k) = g(x(k), u(k)) \tag{4.3}$$

$$x(k) \in \mathbb{X}, \quad u(k) \in \mathbb{U}, \quad k \in \{0, N-1\} \tag{4.4}$$

$$x(N) \in \mathbb{X}_f \subseteq \mathbb{X} \tag{4.5}$$

where Eq. 4.1 is the cost function to minimize, Eq. 4.2 is the difference equation describing the model and initial state, Eq. 4.3 describes how the output is described in terms of the state and input, Eq. 4.4 is the constraints on the states and inputs and Eq. 4.5 is the constraint for the final state. The terminal cost V_f and the terminal constrain set $\mathbb{X}_f$ are often chosen such that closed-loop stability is ensured [14].

4.1.2 Objective

If the models 4.2 and 4.3 are linear, the sets $\mathbb{X}, \mathbb{U}$ and $\mathbb{X}_f$ are defined by affine inequalities and the cost 4.1 is appropriately chosen, i.e. quadratic, the finite horizon optimal control problem results in a quadratic program (QP). A natural choice of cost V_N is

$$V_N = \sum_{i=0}^{N-1} \Big((y(i)-r(i))^T Q (y(i)-r(i)) + u^T(i) R u(i) \Big) + (y(N)-r(N))^T P_f (y(N)-r(N)) \tag{4.6}$$

where r is the reference, Q, R and P_f are positive semi definite weighting matrices. The first summand in 4.6 represents the cost for the output tracking error, the second summand represents the cost for control action and term outside the summation is the terminal output tracking error cost. An advantage with having the finite horizon optimal control problem on QP form is that there are efficient solvers for QP, e.g. solvers provided in the YALMIP toolbox.

4.1.3 Horizon

In the optimization problem 4.6 it assumed that the state prediction horizon and control prediction horizon, denoted by N and M respectively, are equal. In some cases it can be favourable to have $N > M$ and keeping the control signal constant for $M \leq k \leq N$. By having $N > M$ the computational complexity can be reduced since the number of decision variables in the optimization problem is reduced. However, $N > M$ may also decrease the performance of the controller [14].

4.1.4 Linear Time Varying (LTV) MPC

In the case where models 4.2 and 4.3 are nonlinear, the nonlinear dynamics can be approximated by at each time instant k linearize 4.2 and 4.3 around the current state $x(k)$ and the previous control input $u(k-1)$. Then 4.2 can be written as

$$x(k+1) = A_k x(k) + B_k u(k), \quad x(0) = x_0 \tag{4.7}$$

where

$$A_k = \frac{\partial f(x,u)}{\partial x}\Big|_{(x(k),u(k))}, \quad B_k = \frac{\partial f(x,u)}{\partial u}\Big|_{(x(k),u(k))} \tag{4.8}$$

and accordingly in the same way for 4.3

$$y(k) = C_k x(k) + D_k u(k) \tag{4.9}$$

where

$$C_k = \frac{\partial g(x,u)}{\partial x}\Big|_{(x(k),u(k))}, \quad D_k = \frac{\partial g(x,u)}{\partial u}\Big|_{(x(k),u(k))} \tag{4.10}$$

and subscript k emphasizes that matrices varies with time [14] [16]. Worth mentioning is the fact that the nonlinear model 4.2 is linerazed around an operating point

which may not be an equilibrium point. By using the LTV approach the finite horizon optimal control problem for a nonlinear system can be approximated by a QP, which means that the computational burden for solving the optimization problem can be reduced since QP solvers are more efficient than general nonlinear solvers. However, one has to have in mind that at each sampling instant there will be extra computations for calculating the A_k, B_k, C_k and D_k matrices [14].

4.1.5 MPC implementation

Equality constraints

The DBM from section 2.2 is used when predicting the behaviour of the target in the MPC algorithm. The DBM Eq. 2.6-2.10 can be linearized and rewritten as a state space representation with x as a state vector:

$$x = [X \quad Y \quad \dot{x} \quad \dot{y} \quad \psi \quad \dot{\psi}]^T \tag{4.11}$$

and u as control input:

$$u = [a_x \quad \delta_f]^T. \tag{4.12}$$

At each time step when the optimization problem is to be solved the DBM Eq. 2.6-2.10 is first linearized around the current estimate of the state from the EKF. Hence, the system can be expressed on discrete state space form

$$x^+ = Ax + Bu \tag{4.13}$$

where superscript $^+$ denotes the successor state. This LTV state space model can be used as an equality constraint when solving the MPC optimization problem.

Inequality constraints

The input signal has a bounded range. The steering control signal can not exceed the minimum or maximum steering angle and the acceleration control signal cannot exceed the minimum or maximum acceleration. These properties can be seen as an inequality constraint and can be formulated as

$$\begin{bmatrix} a_{min} \\ \delta_{fmin} \end{bmatrix} \leq u \leq \begin{bmatrix} a_{max} \\ \delta_{fmax} \end{bmatrix} \tag{4.14}$$

Additionally, a constraint on how fast the steering angle can change

$$\Delta\delta_f(k) = \delta_f(k) - \delta_f(k-1) \tag{4.15}$$

is included due to the fact that there are limitations on the steering servo of the RC car.

$$\Delta\delta_{f_{min}} \leq \Delta\delta_f \leq \Delta\delta_{f_{max}} \tag{4.16}$$

In this specific case the states are not bounded. Hence, no inequality constraints regarding the states are included.

Objective

The MPC algorithm is repeatedly solving an optimization problem. To solve an optimization problem an objective is needed. In this case the goal is to optimize the trajectory following of the target to the reference trajectory. The objective function is implemented as

$$V_N = \sum_{i=0}^{N} \Big(Cx(i) - y_{ref}(i)\Big)^T Q\Big(Cx(i) - y_{ref}(i)\Big) + u^T(i)Ru(i) \tag{4.17}$$

with the terminal cost $P_f = Q$ and control horizon $M = N$.

The used reference is given by a X and a Y position and a reference yaw angle ψ

$$y_{ref} = \begin{bmatrix} X_{ref} \\ Y_{ref} \\ \psi_{ref} \end{bmatrix} \tag{4.18}$$

and the output matrix is consequently given by

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{4.19}$$

In this simple case we are only interested in minimizing the deviation from the reference trajectory. No other states are taken into consideration and can take whatever value.

Tuning of MPC

In an MPC algorithm there are several tuning parameters, hence extensive tuning is necessary in order to get a desired behavior of the controller. The tuning parameters in the implemented MPC controller are prediction horizon N and the weighting matrices Q and R. Since the dynamics of the RC car differs from the dynamics of a full sized car the tuning of MPC is different for the RC car compared to a full sized car. For the RC car the Q and R matrices are chosen to

$$Q = \begin{bmatrix} 580 & 0 & 0 \\ 0 & 580 & 0 \\ 0 & 0 & 3000 \end{bmatrix} \tag{4.20}$$

$$R = \begin{bmatrix} 100 & 0 \\ 0 & 150 \end{bmatrix} \tag{4.21}$$

and with prediction horizon $N = 10$ and sampling time $T = 0.1\ s$. For a full sized car the Q and R matrices are chosen to

$$Q = \begin{bmatrix} 400 & 0 & 0 \\ 0 & 400 & 0 \\ 0 & 0 & 1.5 \cdot 10^6 \end{bmatrix} \tag{4.22}$$

$$R = \begin{bmatrix} 10 & 0 \\ 0 & 100 \end{bmatrix}. \tag{4.23}$$

The prediction horizon is set to $N = 10$ for the full sized car and sampling time $T = 0.1\ s$. Hence, the prediction horizon is 1 s for both the RC car and the full sized car.

Solver

The YALMIP toolbox is used it comes with a variety of available solvers. The solver used in the thesis is QUADPROG, which is a QP solver. The YALMIP toolbox is easily integrated in Matlab and QUADPROG is available in the MathWorks Optimization toolbox.

4.2 P-controller with feed forward prediction

Another control strategy with less computation requirements is a P-controller with a feed forward prediction. The strategy is to apply a steering to the target proportional to the look-ahead lateral position error.

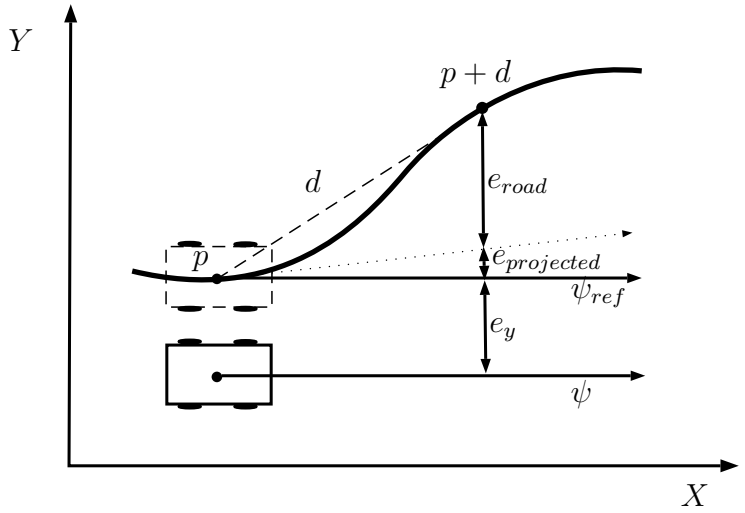


Figure 4.1: P control strategy with lateral look-ahead position error.

4.2.1 Steering Control

Steering

The control law is defined as proportional to the look-ahead lateral path error [17].

$$\delta_f = K_p(e_y + e_{lead}) \tag{4.24}$$

where

$$e_{lead} = e_{projected} + e_{road} \tag{4.25}$$

where e_y is the current lateral error, $e_{projected}$ depends on the current orientation error, e_{road} includes information about the look-ahead reference point and K_p is the

proportional gain.

Since the trajectory following depends on the speed of the vehicle, the feedback gain K_p should be speed dependent to avoid unnecessary steering commands. The value of K_p can be determined analytically by using a curb-following strategy, suggested by [17]. Assume that the rear axis of the vehicle is placed at the reference trajectory. Furthermore, assume that the rear axis of the vehicle has zero slip angle and that it is directed along the reference trajectory. Then one can define, at every time step, a circular arc between the current position of the rear axis and the look-ahead reference point using the current yaw angle [17]. The curvature of the circular arc is given is by $1/R$, where R is the radius of the circle. The curvature of the circular arc can be obtained by the following relation

$$\dot{\psi}_{ref} = \frac{V_x}{R}. \tag{4.26}$$

Assuming the angle between the longitudinal axis of the vehicle and the straight line between the current and look-ahead reference points is small, it follows from Fig. 4.1 that

$$\frac{1}{R} = \frac{2e_{lead}}{d'^2} \tag{4.27}$$

where d' is the distance between the rear axis of the vehicle and the next reference point

$$d' = d + l_r \tag{4.28}$$

and d is the pre-defined distance between the points on the reference trajectory [17]. If the look-ahead distance/time is chosen to short the steering commands will be too aggressive. On the other hand, if the look-ahead distance/time is chosen to long the vehicle will cut corners. Hence, the look-ahead distance/time is something that can be tuned in order to get desired behavior.

In order for the vehicle to follow the curvature of the online defined circular arc, the needed steering angle is

$$\begin{aligned} \delta_f &= \frac{L + K_v V_x}{V_x^2} a_{y_{curb}} = \frac{L + K_v V_x}{V_x^2} \frac{V_x^2}{R} \\ &= \frac{L + K_v V_x^2}{R} = 2\frac{L + K_v V_x^2}{d'^2} e_{lead}. \end{aligned} \tag{4.29}$$

Consequently, the speed dependent gain is given by

$$K_p = 2\frac{L + K_v V_x^2}{d'^2} \tag{4.30}$$

where L is the length of the vehicle and K_v is the under steering factor.

Under steering factor

There are three different scenarios for the under steering factor K_v which depends on the relative values of the cornering stiffness of the front and rear tires and the mass distribution of the vehicle. The three scenarios are neutral, understeer and oversteer. In the neutral case K_v is equal to zero because the slip at the front and rear tires are equal. In the understeering case $K_v > 0$ because there are larger slip on the front tires than on the rear tires. In the oversteering case $K_v < 0$ because there are smaller slip on the front tires than on the rear tires [1]. K_v can be calculated by

$$K_v = \frac{m_f}{2C_f} - \frac{m_r}{2C_r} \tag{4.31}$$

where m_f and m_r are the mass distribution

$$m_f = m\frac{l_f}{L}, \quad m_r = m\frac{l_r}{L}. \tag{4.32}$$

4.2.2 Velocity Control

The look-ahead lateral position error controller assumes constant longitudinal velocity and calculates only the necessary steering command for tracking the reference trajectory. In order to maintain constant velocity of the vehicle a simple control strategy for the acceleration command is implemented

$$a_x = \frac{(\hat{V}_x - V_{x_{ref}})}{T} \tag{4.33}$$

where, $\hat{V}_x$ is the estimated longitudinal velocity from the EKF and $V_{x_{ref}}$ is the desired longitudinal velocity defined by the reference trajectory. Constant velocity is assumed leading to that the desired velocity easily can be calculated from the reference trajectory as

$$V_x = \frac{\sqrt{(X_{ref,k} - X_{ref,k-1})^2 + (Y_{ref,k} - Y_{ref,k-1})^2}}{T} \tag{4.34}$$

5
Communication

To send messages to and from the central server to the different targets wireless communication is a requirement in the framework. The communication part of the framework is presented and discussed in this chapter together with some theory.

5.1 Protocol

The transport control protocol (TCP/IP) and the user datagram protocol (UDP) are the network protocols that are commonly used for transmitting data. TCP/IP is a confirmation based protocol, which means that the transmitter first sends a request to transmit data and awaits an acknowledgment from the receiver, i.e ensures the connection. If connection is ensured, the transmitter sends data to the receiver and awaits another acknowledgment. If no acknowledgment for the data is received, the transmitter assumes that the data is lost and re-transmits the data. Before transmitting the data, it is divided into segments. Each segment has a sequence number to prevent the receiver from decoding the data segments in incorrect order. These error preventing functions may cause unwanted delays in the communication system [18]. Hence, TCP/IP may not be the most suitable protocol for real-time applications [19].

In contrast to TCP, UDP is not confirmation based. Hence, there is no guarantee that the data is correctly received or received at all. On the other hand, since UDP does not have connection check the transmitter can provide the network layer with data at any desired rate. Hence, UDP provides fast but unreliable communication. Worth mentioning is the fact that even though the transmitter can provide the network layer with data in any desired rate, this desired rate may not be the actual end-to-end throughput. Slower end-to-end throughput can be explained by data congestion or limitations on intermediary links [18].

5.2 Latency

We define communication latency of the centralized server setup as the time between sending the control signal from the server and the time the commanded control signal is measured. More particularly this includes sending the signal, the signal being processed on the target, sending command to the low level controller, sending the measured control signal back to the server and processing the received signal.

5.3 Implementation/communication setup

Since the host computer will run MATLAB the communication between the system and the targets will also be via MATLAB. The communication protocol used will be UDP since there are no severe consequences if a single message is not received correctly. Furthermore, delays caused by not receiving acknowledgement from the receiver is avoided when using the UDP protocol. The communication setup is used with a standard UDP block in Simulink. The message to the RC car is an Extensible Markup Language (XML) message which is converted to uint8 and sent as a UDP message. The messages are sent between the server and the RC car via radio. The RC car receives a XML message consisting of commanded actuator signal or a reference trajectory point depending on the setup. In the researched centralized setup the XML message consists of a steering angle $[rad]$ and a velocity $[m/s]$ command. The reason why velocity is used instead of longitudinal acceleration is that the low level controller on the RC car takes a velocity as an input. To obtain a velocity, the calculated acceleration command is integrated. The RC car processes the message and the low level controller performs the control action. The XML message is constructed in Matlab and encoded to an uint8 to be sent via the UDP Simulink send block.

The control server receives XML messages containing status of the RC car. The status messages includes sensor readings from the GPS, accelerometer, gyroscope and magnetometer. Furthermore, the status message includes actual steering angle and time stamp. An overall illustration of the communication setup is shown in Fig. 5.1.

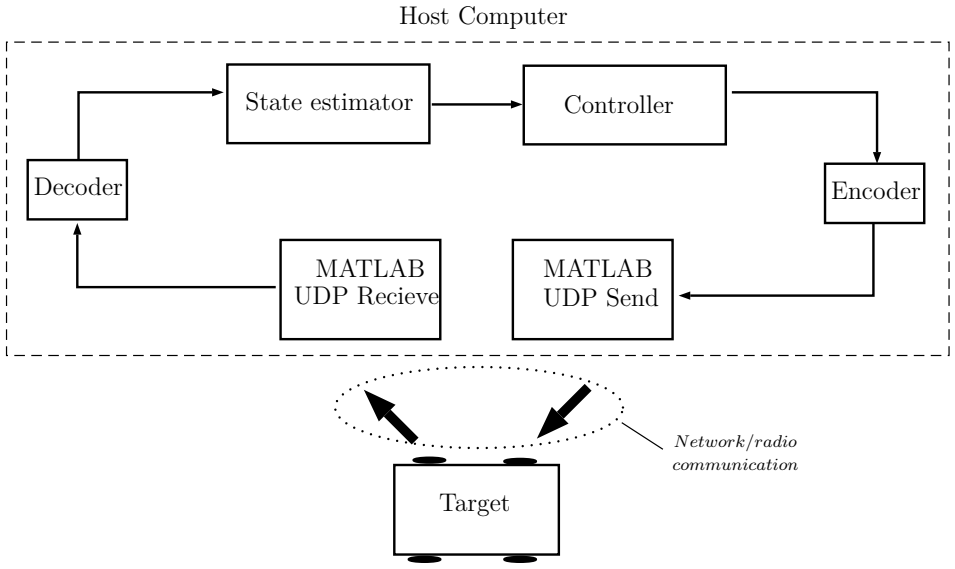


Figure 5.1: Communication setup

The control server for the test targets is sampled with sample time $T = 0.1$ s. Because of the choice of sample time and the fact that the messages only contains some doubles the requirements on the data rate of the communication setup is

relatively low. Hence the proposed setup is not limited by the data rate of the current communication channel.

6
Results

In this chapter the results are presented. First simulation results of the different controllers are shown followed by a comparison. Lastly brief results from communication experiments are shown.

6.1 Simulation setup

To test the controllers from Chapter 4 simulations were performed, presented in this chapter. The chapter is divided into four sections. The first two sections presents the results on the MPC control and the last two sections are handling the results regarding the P control. With the goal to be able to create a versatile framework which can control many different targets simulations were performed on both a full sized car model and on the smaller RC car. In the simulations the DBM from section 2.2 is used together with the parameters in Table 2.2 when simulating the behaviour of the full sized car and when simulating the behaviour of the RC car the identified parameters in Table 2.1 is used together with the DBM. To test the controllers performance characteristics they were tested on two different scenarios:

- **Scenario 1:** A straight trajectory but the initial position has an offset to the reference, to illustrate how good the controller converges towards the reference.
- **Scenario 2:** A trajectory consisting of a straight segment followed by four curves and finally an additional straight segment, to illustrate how well the controllers manage steering actions.

In all simulations a sampling time of $T = 0.1\ s$ is used. The velocity is varying in the simulations depending on the scenario and car type.

6.2 MPC - Full sized car simulations

The implemented MPC controller presented in section 4.1.5 are tested on the two different scenarios. When evaluating the MPC controller for a full sized car, simulations were made to test the controller in section 4.1.5 with the tuning values from Eq. 4.22 and 4.23.

Scenario 1

In Fig. 6.1-6.5 the simulation results, simulating the first scenario, is presented. It can be observed in Fig. 6.1 that the steady state error converges to zero from the initial lateral error of 1 m. The yaw angle is illustrated in Fig. 6.2. From the figure it can be concluded that the yaw angle also converges towards a zero error to the yaw angle reference.

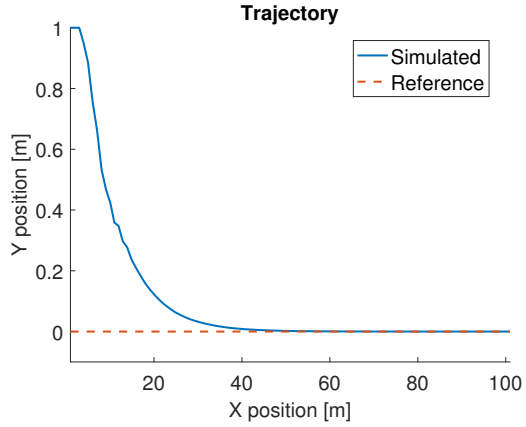


Figure 6.1: Straight line MPC control trajectory for full sized car.

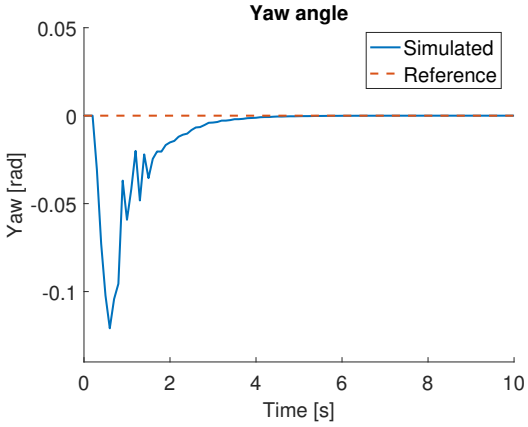


Figure 6.2: Straight line MPC control yaw angle (ψ) for full sized car.

In Fig. 6.3 it can be seen that the penalty on the acceleration in the objective function, Eq. 4.23, leads to a smoother longitudinal velocity. The lateral velocity seen in Fig. 6.4 is fluctuating sligthly more due to the steering control seen in Fig. 6.5.

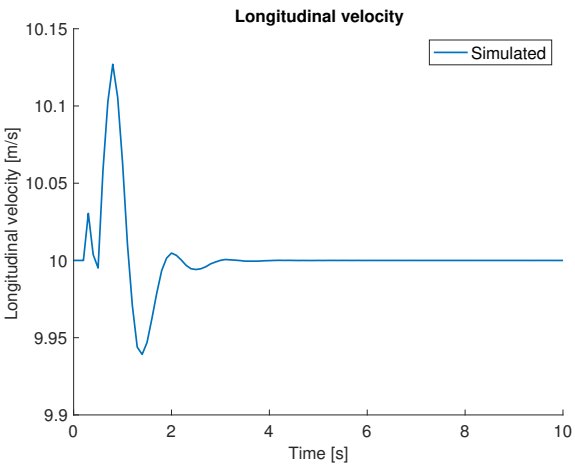


Figure 6.3: Straight line MPC control longitudinal velocity ($\dot{x}$) for full sized car.

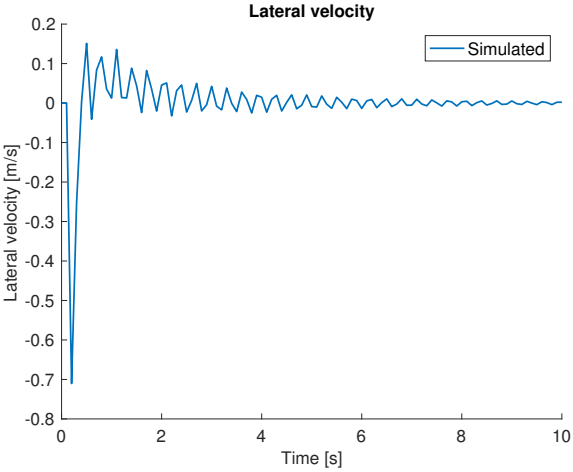


Figure 6.4: Straight line MPC control lateral velocity ($\dot{y}$) for full sized car.

In Fig. 6.5 it can be seen that the steering, blue line, is well within the constraints, black dotted lines.

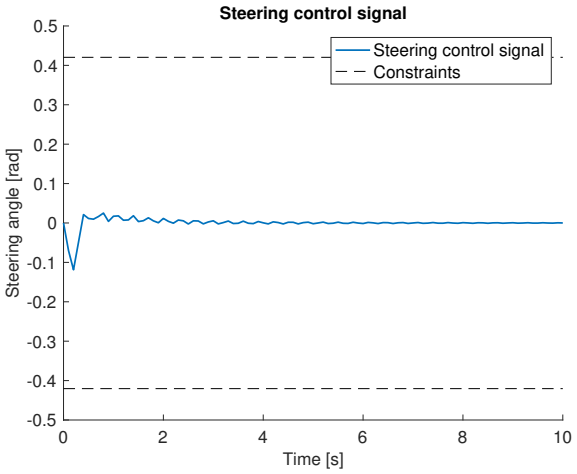


Figure 6.5: Straight line MPC control steering angle (δ_f) for full sized car.

Scenario 2

The same controller was likewise tested simulating scenario 2. In Fig. 6.6-6.11 results from simulations of scenario 2 is presented. It can be seen in Fig. 6.6 that the controller follows the trajectory well and the same thing can be said about the yaw angle, see Fig. 6.7.

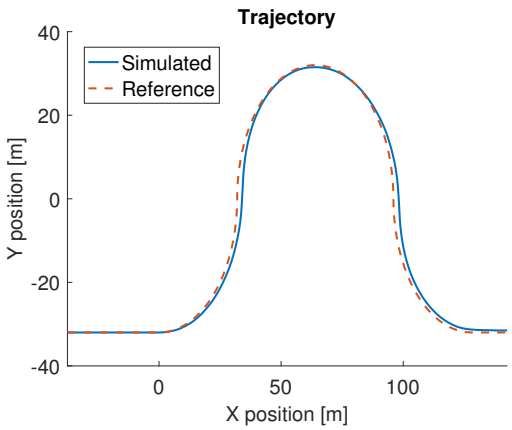


Figure 6.6: MPC control trajectory for full sized car.

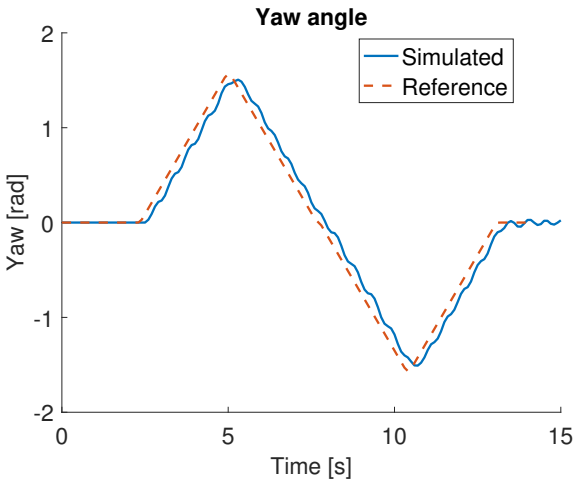


Figure 6.7: MPC control yaw angle (ψ) for full sized car.

The penalty on longitudinal acceleration leads to a smooth longitudinal velocity seen in Fig. 6.8.

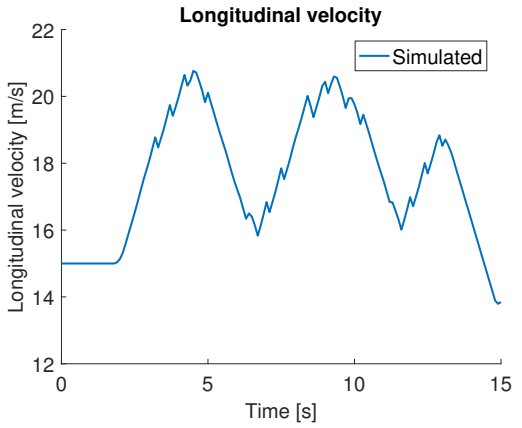


Figure 6.8: MPC control longitudinal velocity ($\dot{x}$) for full sized car.

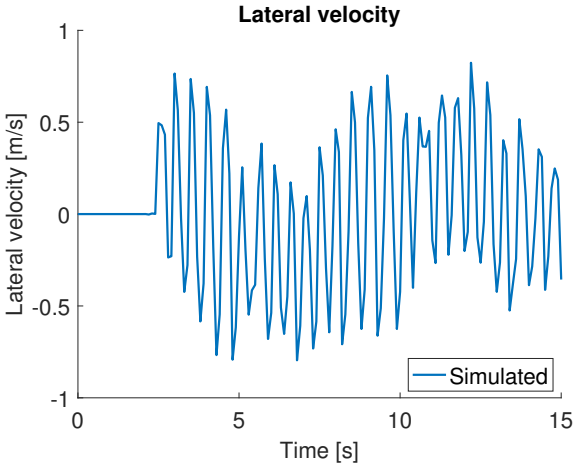


Figure 6.9: MPC control lateral velocity ($\dot{y}$) for full sized car.

The lateral error, Fig. 6.10 is relatively large and reaches almost 2 m at the most. It can be seen that the steering control signal, Fig. 6.11, tries to decrease the lateral position error but also has an high cost in the objective function which makes the steering stay rather low.

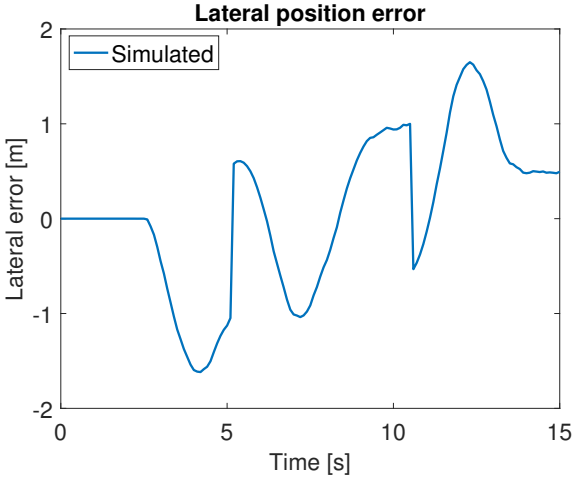


Figure 6.10: Straight line MPC control lateral error (e_y) for full sized car.

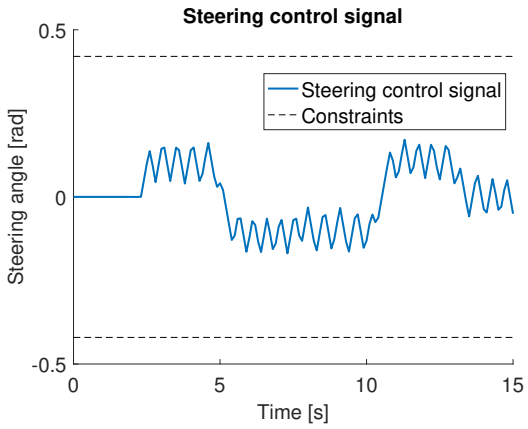


Figure 6.11: Straight line MPC control steering angle (δ_f) for full sized car.

6.3 MPC - RC car simulations

The controller from Section 4.1.5 with the tuning values from Eq. 4.20 and 4.21 was also tested in simulations with the model from 2.2 and the identified values from Table 2.1 to simulate the behaviour of the RC car. The RC car is simulated on the two same scenario as the full sized car in Section 6.2 with some modifications to fit the different dynamics of the RC car compared to a full size car. The reference is instead structured with a velocity of 5 m/s. Furthermore the initial position offset of the RC car is instead set to 0.5 m to match the new velocity.

Scenario 1

Fig 6.12-6.16 illustrates the results from the first scenario. In Fig. 6.12 it can be seen that the controller converges towards a zero lateral position error. The yaw angle is converging but keeps oscillating around the reference quite long before converging towards the reference, see Fig. 6.13

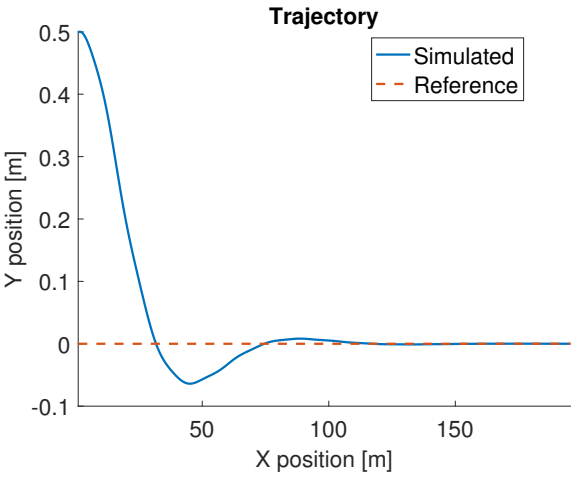


Figure 6.12: Straight line MPC control trajectory for RC car.

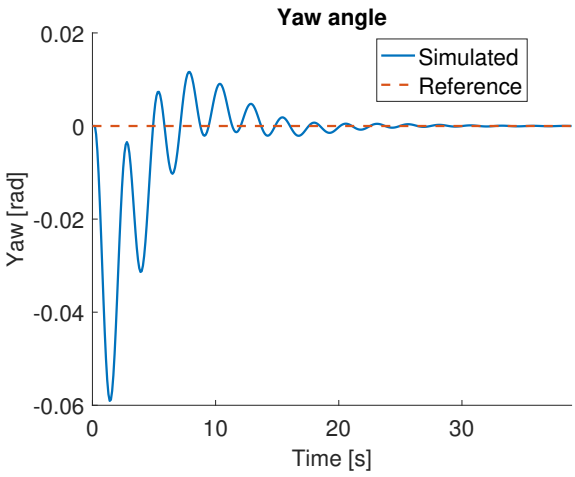


Figure 6.13: Straight line MPC control yaw angle (ψ) for RC car.

It can be observed in Fig. 6.14 that the longitudinal velocity is almost constant during the entire simulation. Fig. 6.15 shows that the lateral velocity converges to zero, which coincides with the results presented in Fig. 6.12 and 6.13.

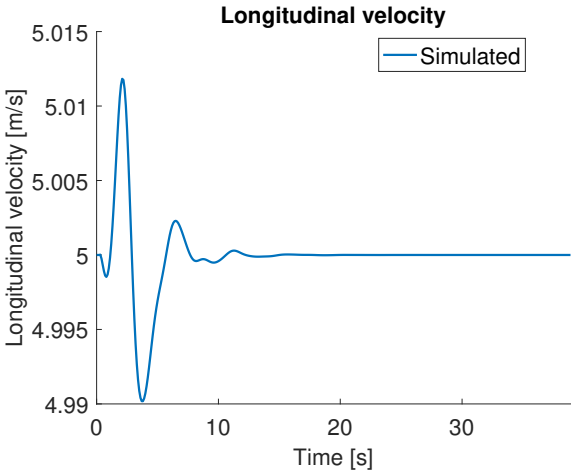


Figure 6.14: Straight line MPC control longitudinal velocity ($\dot{x}$) for RC car.

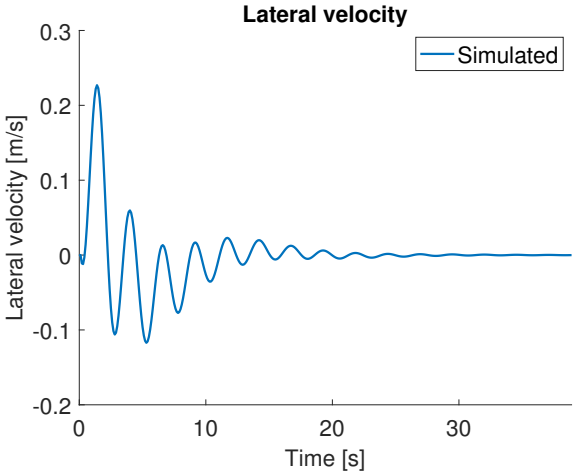


Figure 6.15: Straight line MPC control lateral velocity ($\dot{y}$) for RC car.

The steering control signal is oscillating around zero but converges with time, see Fig. 6.16.

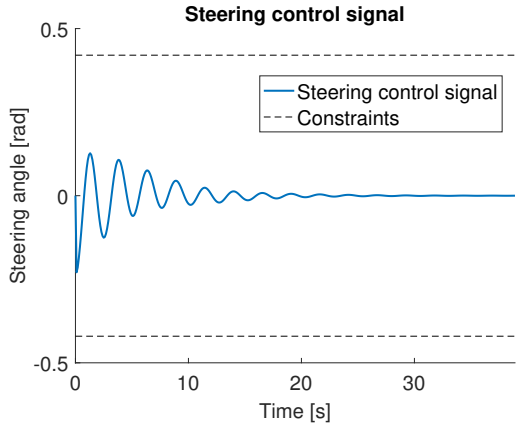


Figure 6.16: Straight line MPC control steering command (δ_f) for RC car.

Scenario 2

The results from simulations of the RC car with scenario 2 are found in Fig. 6.17-6.22. In Fig. 6.17 it can be seen that the MPC deviates slightly from the trajectory. However, the yaw reference is tracked properly, see Fig. 6.18.

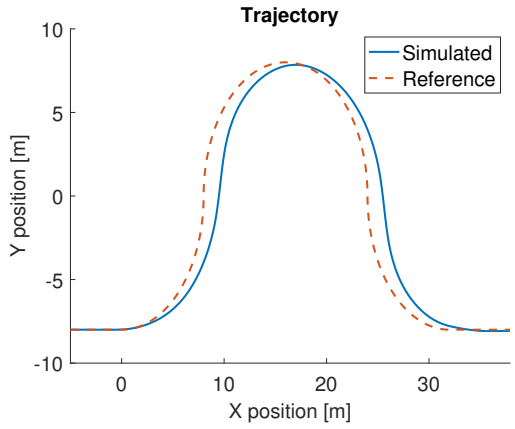


Figure 6.17: MPC control trajectory for RC car.

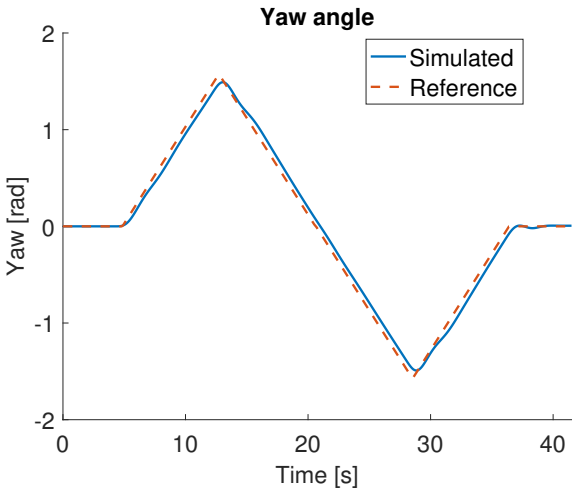


Figure 6.18: MPC control yaw angle (ψ) for RC car.

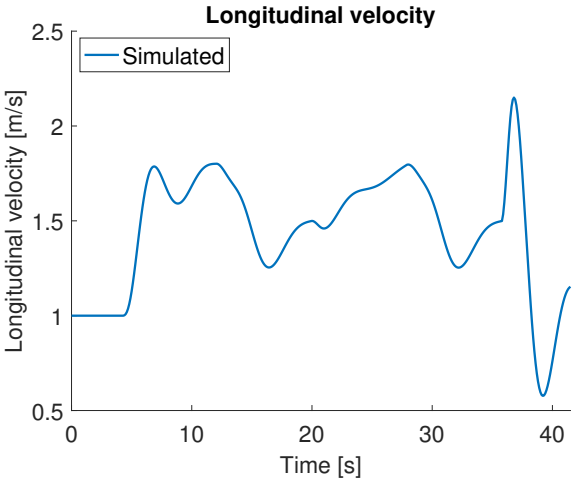


Figure 6.19: MPC control longitudinal velocity ($\dot{x}$) for RC car.

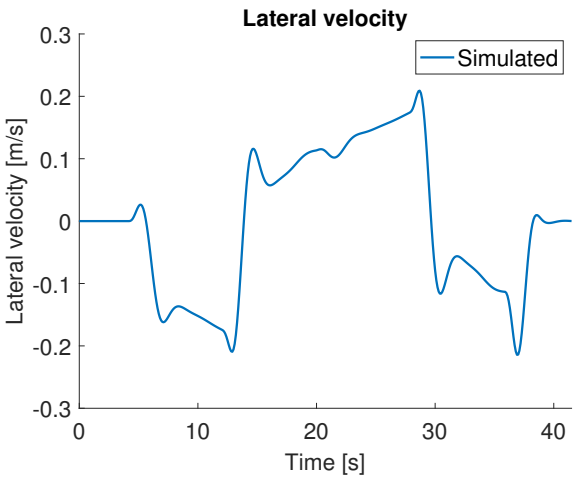


Figure 6.20: MPC control lateral velocity ($\dot{y}$) for RC car.

In Fig. 6.21 it can be seen that the maximum lateral error is roughly 0.5 m. In Fig. 6.22 it can be seen that the steering angle fluctuates but also that is stays inside the constraints.

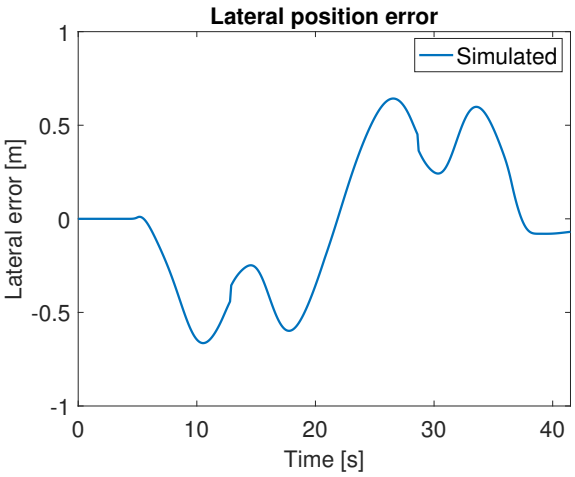


Figure 6.21: MPC control lateral error (e_y) for RC car.

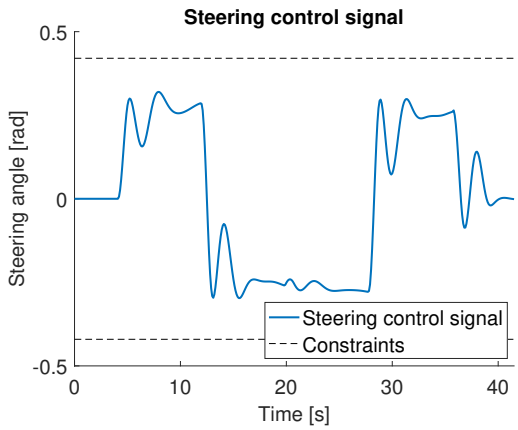


Figure 6.22: MPC control steering angle (δ_f) for RC car.

6.4 P control - Full sized car simulations

The implemented P controller with a feed forward term presented in section 4.2 is tested on the two scenarios. When evaluating the P controller simulations were made to test the controller in section 4.2. To simulate the behavior of a full size car the model from section 2.2 were used together with the parameter values found in Table 2.2. A sampling time of $T = 0.1$ s was used and the look-ahead time was tuned to 0.2 s. In Fig. 6.23-6.27 the simulation results, simulating the first scenario, are presented.

Scenario 1

The P controller drives the lateral error to zero in about 3 s, which can be seen in Fig 6.23. The yaw angle, in Fig. 6.24 is fluctuating due to the oscillating steering control from the P controller, see Fig. 6.27, but is then driven to the reference angle.

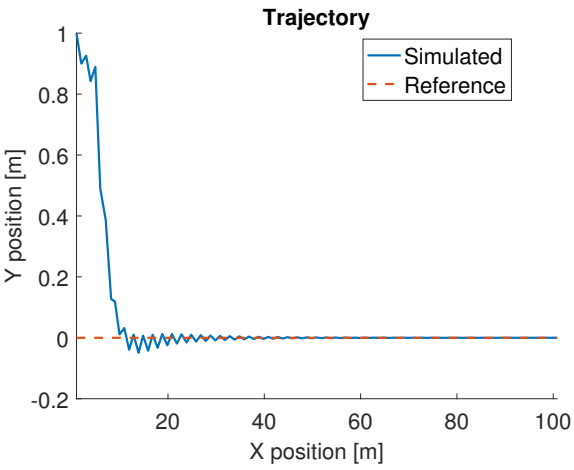


Figure 6.23: Straight line P control trajectory for full sized car.

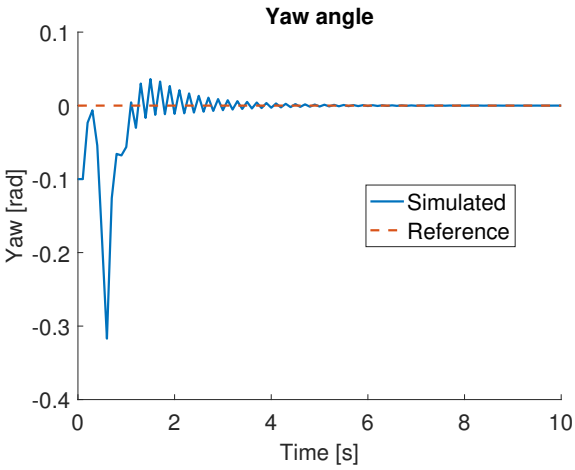


Figure 6.24: Straight line P control yaw angle (ψ) for full sized car.

The P controller only controls the steering. The longitudinal acceleration is controlled as in Section. 4.2.2 leading to the oscillating longitudinal velocity in 6.25. The lateral velocity is oscillating due to the steering, see Fig. 6.26 and 6.27.

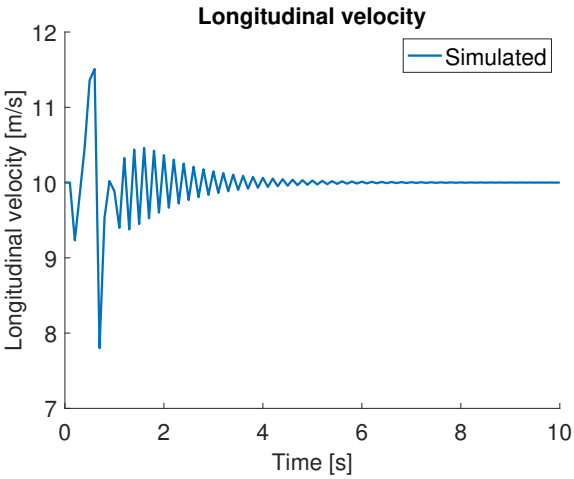


Figure 6.25: Straight line P control longitudinal velocity ($\dot{x}$) for full sized car.

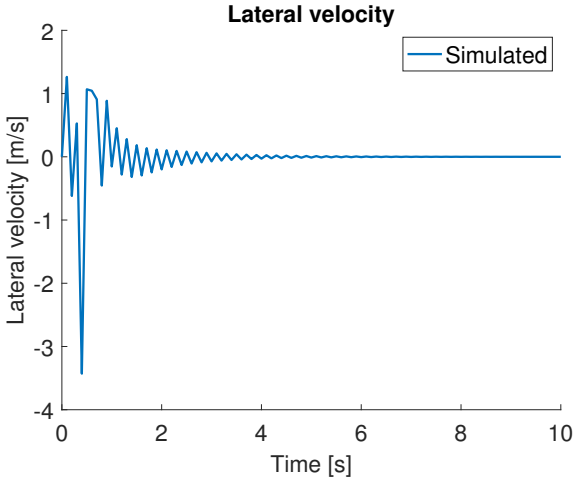


Figure 6.26: Straight line P control lateral velocity ($\dot{y}$) for full sized car.

The steering control signal is low with small changes in the start and then converging towards zero.

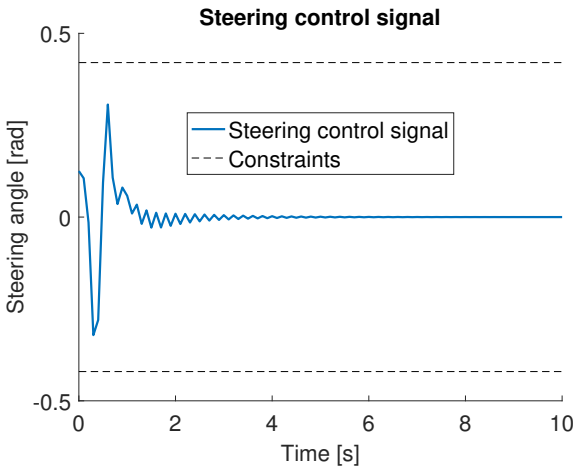


Figure 6.27: Straight line P control steering command (δ_f) for full sized car.

Scenario 2

In Fig. 6.28-6.33 the results from simulating scenario 2 can be seen.
Fig. 6.28 shows that the P controller follows the trajectory successfully. However in Fig. 6.29 it can be seen that the yaw angle slightly fluctuates around the reference.

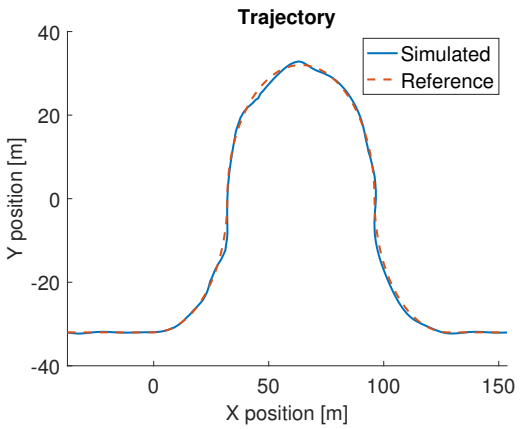


Figure 6.28: P control curve trajectory for full sized car.

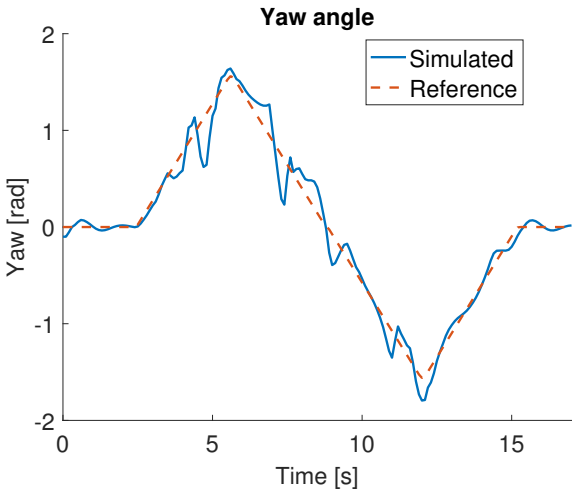


Figure 6.29: P control yaw angle (ψ) for full sized car.

It can be seen in Fig. 6.30 that the longitudinal velocity varies quite much, which indicates that the velocity control strategy presented in Section 4.2.2 might be too naive. In Fig. 6.31 it can be seen that the lateral velocity is close to zero during most time of the simulation but that there are some larger peaks and drops. Furthermore it can be observed by comparing Fig. 6.31 and 6.33 that the larger peaks and drops in lateral velocity occurs when it is larger peak or drop in the steering angle control signal.

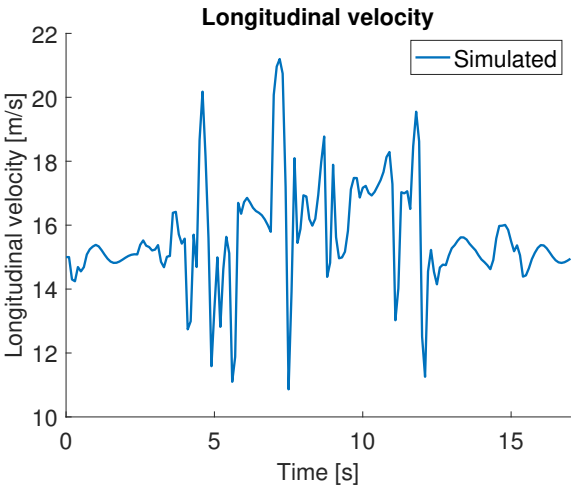


Figure 6.30: P control longitudinal velocity ($\dot{x}$) for full sized car.

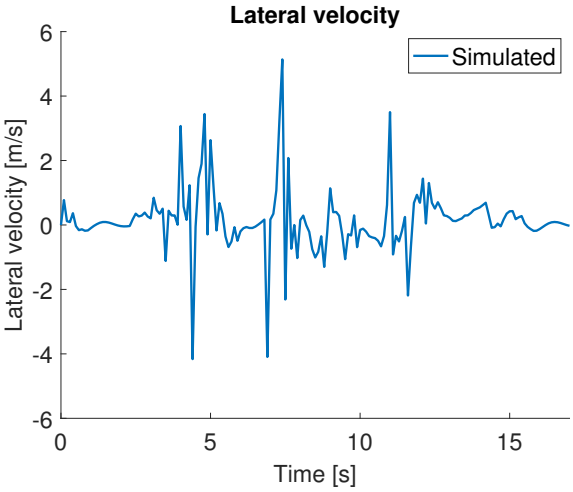


Figure 6.31: P control lateral velocity ($\dot{y}$) for full sized car.

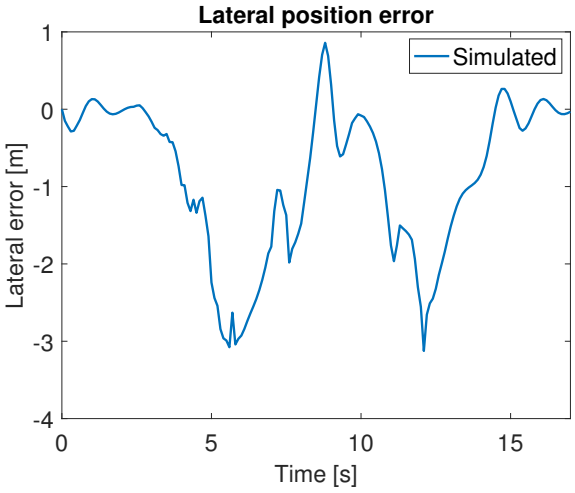


Figure 6.32: P control lateral error (e_y) for full sized car.

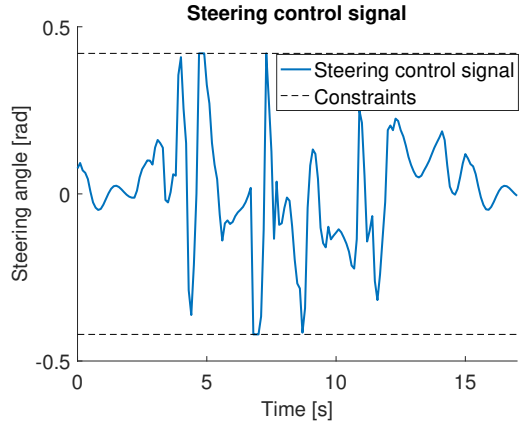


Figure 6.33: P control steering angle (δ_f) for full sized car.

6.5 P control - RC car simulations

The controller from Section 4.2 was also tested in simulations with the model from Section 2.2 and the identified values from Table 2.1 to simulate the behaviour of the RC car. The P controller was tested for the two scenarios and the look-ahead time was tuned to 1 s.

Scenario 1

The results from the simulations of the first scenario are illustrated in Fig. 6.34-6.38. The P controller drives the lateral error to zero but with an oscillating behaviour, seen in Fig. 6.34. Furthermore, it can be observed in Fig. 6.34 and 6.35 that tracking error for the trajectory and yaw angle converges slowly to zero.

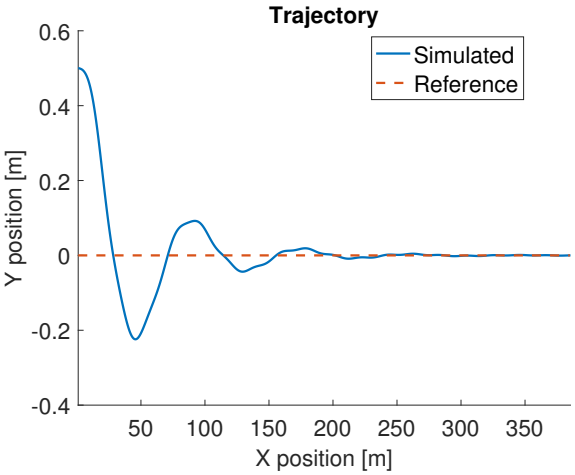


Figure 6.34: Straight line P control trajectory for RC car.

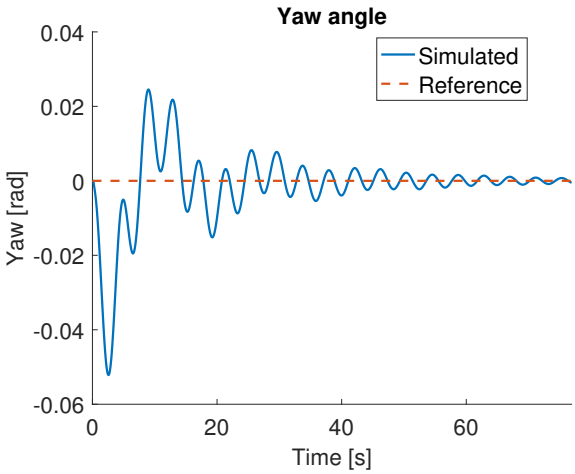


Figure 6.35: Straight line P control yaw angle (ψ) for RC car.

Both the longitudinal and lateral velocities oscillates slightly around 5 m/s and 0 m/s respectively, during the the entire simulation, see Fig. 6.36 and 6.37.

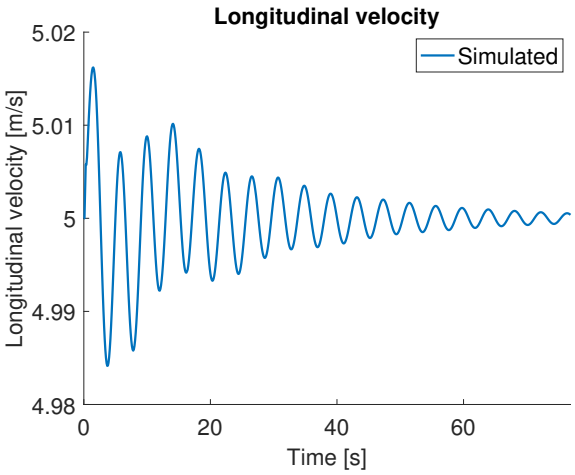


Figure 6.36: Straight line P control longitudinal velocity ($\dot{x}$) for RC car.

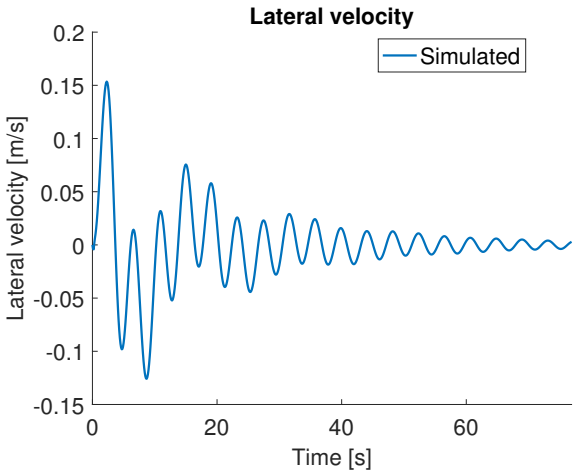


Figure 6.37: Straight line P control lateral velocity ($\dot{y}$) for RC car.

The steering angle has small changes in the beginning and is then driven towards zero, see Fig. 6.38.

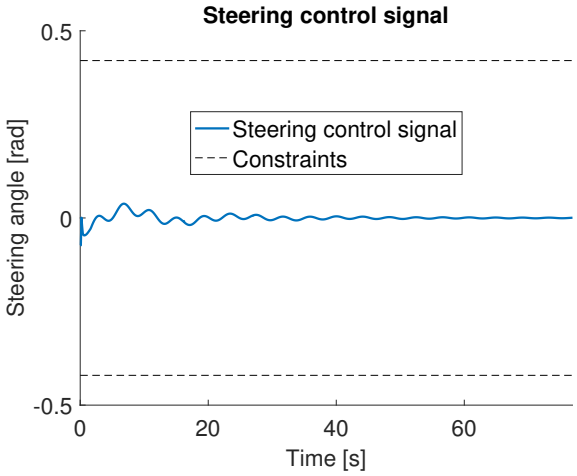


Figure 6.38: Straight line P control steering command (δ_f) for RC car.

Scenario 2

In Fig. 6.39-6.44 results from simulations of the P controller simulating the RC car with scenario 2 can be seen. Fig. 6.39 shows the trajectory and that the P controller does a good job in controlling the model with some small deviations. The yaw angle, Fig. 6.40, is controlled nicely with some minor deviations.

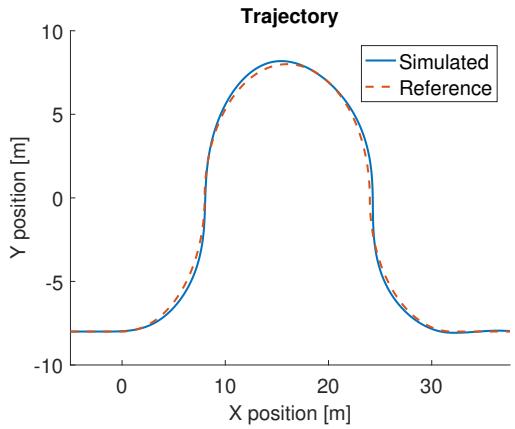


Figure 6.39: P control trajectory for RC car.

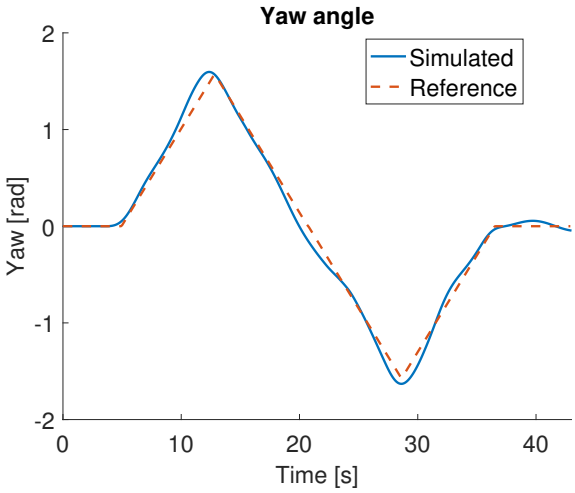


Figure 6.40: P control yaw angle (ψ) for RC car.

In Fig. 6.41 it can be seen that the controller keeps constant velocity in the straight sections and a slightly higher somewhat constant velocity in the curve.

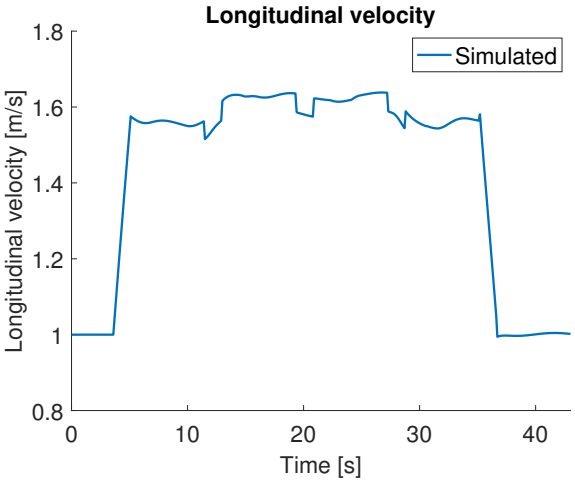


Figure 6.41: P control longitudinal velocity ($\dot{x}$) for RC car.

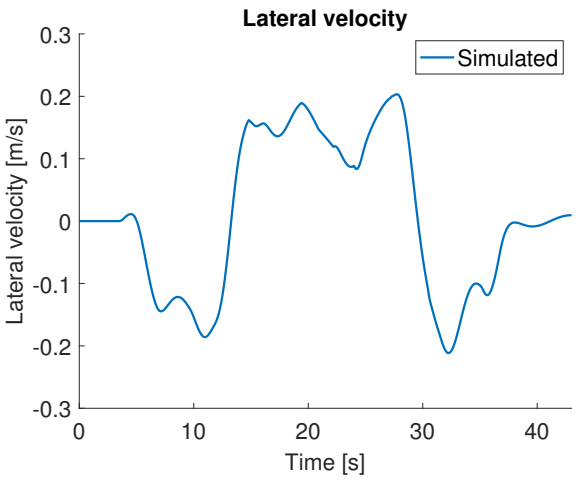


Figure 6.42: P control lateral velocity ($\dot{y}$) for RC car.

The lateral position error is small and is most of the time not more of than 20 cm, see Fig. 6.43. The steering angle is rather smooth and is within the constraints, see Fig. 6.44.

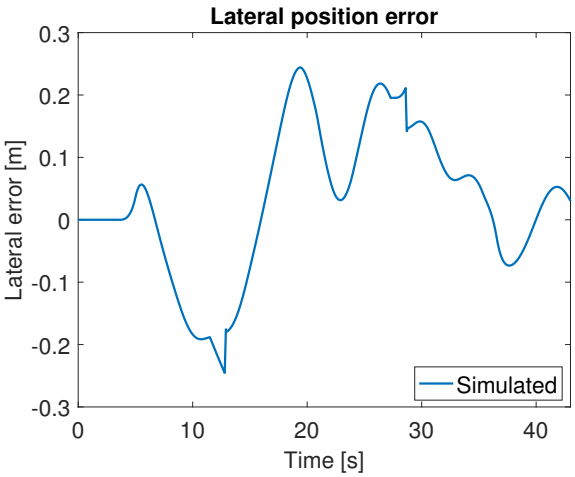


Figure 6.43: P control lateral error (e_y) for RC car.

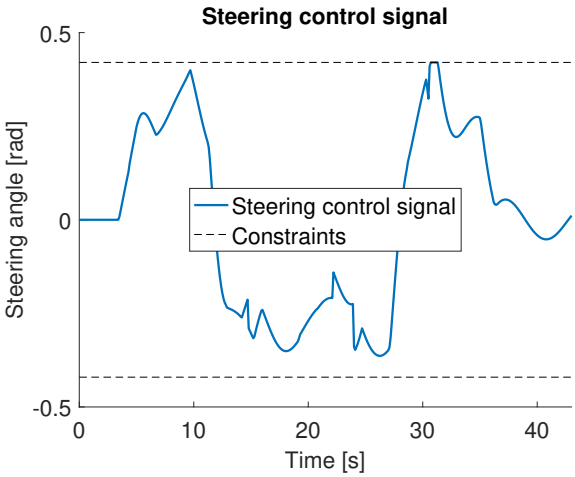


Figure 6.44: P control steering command (δ_f) for RC car.

6.6 Controller comparison

In this section the results for the MPC and P control simulations for full sized car and RC car are summarized and discussed. The controllers are compared with respect to computational complexity, lateral error, yaw error and steady state convergence time.

In Table 6.1 the computation times of the two different controllers are presented.

Table 6.1: Controller computation time

	P control mean computation time [s]	MPC mean computation time [s]
Full sized vehicle	$1.2715 \cdot 10^{-5}$	0.4937
RC car	$8.9098 \cdot 10^{-6}$	0.4223

It can be seen in Table 6.1 that the P controller is significantly faster than the MPC controller. Hence, the P controller is more suitable for a real time implementation.

Results of the lateral position error, of the vehicle with respect to the reference, from scenario 2 is presented in Table 6.2.

Table 6.2: Controller mean absolute lateral error for scenario 2.

Scenario 2	P control mean absolute lateral error [m]	MPC mean absolute lateral error [m]
Full sized vehicle	0.351	0.652
RC car	0.101	0.238

It can be seen that the P controller has smaller mean absolute lateral error than the MPC controller. However, since the MPC controller is only investigated in the simulation environment the tuning of the MPC is less extensive than the tuning of the P controller, which affects the simulation results for the mean absolute lateral error.

Table 6.3: Controller mean absolute yaw error for scenario 2.

Scenario 2	P control mean absolute yaw error [rad]	MPC mean absolute yaw error [rad]
Full sized vehicle	0.0833	0.0539
RC car	0.0809	0.0396

In Table 6.3 it can be seen that the MPC tracks the yaw reference more accurately for both full sized car and RC car. Smaller yaw error can be explained by the fact that the penalty for deviation from yaw reference are substantial for both full sized car and RC car, see Eq. 4.20 and 4.22.

Results from scenario 1 illustrating how fast the controller drives the error to zero is concluded in Table 6.4.

Table 6.4: Controller convergence time

Scenario 1	P control convergence time [s]	MPC convergence time [s]
Full sized vehicle	2.8	5.3
RC car	38.3	23.4

It can be seen in Table 6.4 that the MPC controller converges to zero tracking error slower than the P controller for a full sized vehicle but faster for a RC car.

6.7 Communication - Results

To measure the latency of the centralized server set up experiments are run with the RC car and the time between sending the control signal and the time the commanded control signal is measured is recorded. The results are shown in table 6.5.

Table 6.5: RC car latency

Mean latency	Median latency	Max latency	Min latency
0.0968 s	0.1 s	0.14 s	0.04 s

There is a limitation on how fast measurements can be polled from the RC car. The fastest possible polling time is 11 ms and the polling time used in the experiments was 20 ms. The choice of polling time 20 ms might affect the accuracy slightly but it gives an upper bound on the latency.

7
Discussion

In this chapter the results presented in this thesis are discussed. The discussion focuses on the reasons why the results turned out in the way they did. Furthermore, reasoning about how the limitations affected the results and possible approaches to the solve the remaining problems is done.

7.1 Modelling

Open loop comparison of DMB with identified parameters and measurements for the RC car, presented in Section 2.3.2, shows that there is a slight offset in the identified model. The offset gives an indication that the data set is not good enough. The quality of the data set strongly depends on the available sensors. Another approach for modelling the test targets could be to use Black box modelling, such as ARX or ARMAX models, instead. However, the benefit of having the same physical model for all test targets with just different parameter values is lost.

7.2 EKF

Since the processes model used in the EKF is the DBM with the identified parameters the model uncertainty is introduced in the state estimation as well. Furthermore, the inaccuracy of the available sensors on the RC car brings more uncertainty into the estimation. The GPS and gyroscope are accurate enough but the magnetometer contains a lot of noise. The reason why we use the magnetometer for measuring the yaw angle, even though it is noisy, is that it is the only available sensor which provides information of absolute orientation. To be able to control a test target an accurate estimation of the yaw angle is crucial. The main challenge of implementing an EKF is, in this case, to get a reliable validation method since the ground truth is unknown.

7.3 Results

Simulations showed good results in controlling the targets along the trajectories, both using MPC strategy and P control with feedforward prediction strategy. Additionally, simulations indicated that the computational complexity of the implemented MPC was too high to meet the real time constraints. However, ways to

improve the computational time of the MPC has not been investigated in this thesis.

When performing experimental results with the RC car the controller was unable to make the RC car follow the desired trajectories. The lack of success in controlling the RC car, is probably due to mismatch in the model parameters as well as tuning of the EKF. If no good estimate of the target is available, it is impossible to get a good control.

The P controller shows promising results in Chapter 6 in simulations. These results shows that it is probably a good choice of control strategy, if one were get a more well matching model and/or a more accurate state estimation.

Investigations in whether the control setup should be centralized or decentralized showed that the latency present in the communication system were not to severe for an centralized control setup. However, the research was limited to just implementation of a centralized control setup. Furthermore, the research was also limited to scenarios including only one test target to control.

8

Conclusion

It can be concluded that the DBM is a suitable model for test targets of different size since only the yaw inertia and tire cornering stiffness needs to be identified using system identification. However, to get accurate trajectory following it is of high importance to get an accurate estimation of the parameters, which put demands on the sensors available on the test targets when collecting measurements for system identification.

A MPC control strategy is supposed to be better suited for more complex scenarios due to the more complex and longer prediction. A MPC control strategy uses the model to predict the future behaviour. This makes an accurate prediction which is used to calculate the optimal control signal. MPC can control more than one variable which in this case leads to a smoother velocity control. However the MPC demands more extensive tuning to get a good control. In this thesis this is not the case and it can be seen that the P controller is not only faster but gives more accurate trajectory following in terms of lateral position error, 0.351 m compared to 0.652 m for full sized car and 0.101 m compared to 0.238 m for RC car.

The computing time for the controllers is a key when deciding on a control strategy. The system will need to be run in real time. The P control computes the control signal in $1.2715 \cdot 10^{-5}s$ while the MPC takes $0.4937s$. It can be concluded that the P controller easily meets the real time constraints in the framework but that the MPC controller does not. For an MPC control algorithm to run under real time constraints the computing power would need to be immensely enhanced.

The results from the researched central setup in communication and control shows that the latency observed in the communication is rather small. Given the fact that the steering in the RC car specifically runs at 50 Hz the latency of 0.0968 s seen in Tab. 6.5 The communication has no big impact on the choice of centralized vs. decentralized control setup.

However it can be predicted that the computing complexity is enhanced with the number of test targets in a scenario. For further work a decentralized framework could be researched in order to run more complex scenarios with increasing numbers of actors. We also suggest future work can include research in whether a more sophisticated control strategy than the one presented in Section 4.2.2 can be implemented for the velocity control. Furthermore, it can be investigated if it is possible to get a more accurate identification of the DBM parameters of the RC car.

Bibliography

[1] Rajesh Rajamani. *Vehicle Dynamics and Control*. 2012.

[2] Riksdagen. Nollvisionen och det trafiksäkra samhället Proposition 1996/97:137 - Riksdagen.

[3] Volvo car corporation. Vision | Volvo Car Group.

[4] L. Caracciolo, a. de Luca, and S. Iannitti. Trajectory tracking control of a four-wheel differentially driven mobile robot. *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, 4(May):2632–2638, 1999.

[5] M. Schorn, U. Stahlin, a. Khanafer, and R. Isermann. Nonlinear trajectory following control for automatic steering of a collision avoiding vehicle. *2006 American Control Conference*, pages 5837–5842, 2006.

[6] M.C. Best, T.J. Gordon, and P.J. Dixon. An Extended Adaptive Kalman Filter for Real-time State Estimation of Vehicle Handling Dynamics. *Vehicle System Dynamics*, 34(1):57–75, 2000.

[7] Yiqun Dong, Youmin Zhang, and Jianliang Ai. Experimental Test of Artificial Potential Field-Based Automobiles Automated Perpendicular Parking. *International Journal of Vehicular Technology*, 2016, 2016.

[8] Xianghui Yuan, Feng Lian, and Chongzhao Han. Models and algorithms for tracking target with coordinated turn motion. *Mathematical Problems in Engineering*, 2014, 2014.

[9] Johnny Ngu and Ronakorn Soponpunth. Fusing data from multiple vehicles into a common picture. 2016.

[10] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and Dynamic Vehicle Models for Autonomous Driving Control Design. *IEEE Intelligent Vehicles Symposium*, (Iv):1094–1099, 2015.

[11] Romain Pepy, Alain Lambert, and Hugues Mounier. Path Planning using a Dynamic Vehicle Model. *Information and Communication Technologies, ICTTA 06*, 1(1):781 – 786, 2006.

[12] Lennart Ljung. System Identification, Report no. LiTH-ISY-R-2809. 2007.

[13] Simo Särkkä. Bayesian filtering and smoothing. *Bayesian Filtering and Smoothing*, pages 1–232, 2010.

[14] P Falcone, F Borrelli, J Asgari, H E Tseng, and D Hrovat. Predictive Active Steering Control for Autonomous Vehicle Systems. *Control Systems Technology, IEEE Transactions on*, 15(3):566–580, 2007.

[15] D.Q.˜Mayne, J.B.˜Rawllings, C.V.˜Rao, and P.O.M.˜Skokaert. Cons\-trai\-ned model predictive control: {S}tability and optimality. *Automatica*, 36(6):789–814, 2000.

[16] Alexander Katriniok and Dirk Abel. LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics. *Proceedings of the IEEE Conference on Decision and Control*, pages 6828–6833, 2011.

[17] George Mavros, Homer Rahnejat, and Paul King. Analysis of the transient handling properties of a tyre, based on the coupling of a flexible carcass - Belt model with a separate tread incorporating transient viscoelastic frictional properties. *Vehicle System Dynamics*, 43(SUPPL.):199–208, 2005.

[18] Keith (Polytechnic Institute of NYU) Ross and Amherst) Kurose, James (University of Massachusetts. *Computer Networking*. Number 6. 2013.

[19] Saghir Munir and Wayne J. Book. Internet-based teleoperation using wave variables with prediction. *IEEE/ASME Transactions on Mechatronics*, 7(2):124–133, 2002.