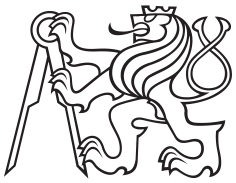
Master Thesis



Czech
Technical
University
in Prague

F3

Faculty of Electrical Engineering
Department of Cybernetics

MPC Based Control Algorithms for Vehicle Control

Vít Cibulka

Supervisor: Ing. Tomáš Haniš, Ph.D.
Field of study: Robotics
Subfield: Cybernetics and Robotics
May 2019

MASTER'S THESIS ASSIGNMENT

I. Personal and study details

Student's name: **Cibulka Vít** Personal ID number: **420267**

Faculty / Institute: **Faculty of Electrical Engineering**

Department / Institute: **Department of Cybernetics**

Study program: **Cybernetics and Robotics**

Branch of study: **Robotics**

II. Master's thesis details

Master's thesis title in English:

MPC Based Control Algorithms for Vehicle Control

Master's thesis title in Czech:

Řídicí systém dynamiky vozu založený na MPC algoritmech

Guidelines:

The goal of the thesis is to investigate control algorithms for land vehicles in order to develop full-time and full-authority control system. Model Predictive Control methodology will be employed in order to handle system hard constrains. The thesis will be done in following steps:
1. Develop non-linear verification mathematical model.
2. Derivation of design model suitable for MPC algorithms.
3. Control system problem formulation suitable for MPC methodology.
4. Implemantation of MPC algorithm in Matlab & Simulink.
5. Control algorithms verificaiton.

Bibliography / sources:

[1] Dieter Schramm, Manfred Hiller, Roberto Bardini – Vehicle Dynamics – Duisburg 2014
[2] Hans B. Pacejka - Tire and Vehicle Dynamics – The Netherlands 2012
[3] Milan Korda, Igor Mezic - Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control - 2018
[4] William F. Milliken, Douglas L. Milliken – Race Car Vehicle Dynamics – Warrendale 1995

Name and workplace of master's thesis supervisor:

Ing. Tomáš Haniš, Ph.D., Department of Control Engineering, FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **24.01.2019** Deadline for master's thesis submission: **24.05.2019**

Assignment valid until: **30.09.2020**

Ing. Tomáš Haniš, Ph.D.	doc. Ing. Tomáš Svoboda, Ph.D.	prof. Ing. Pavel Ripka, CSc.
Supervisor's signature	Head of department's signature	Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

.	
Date of assignment receipt	Student's signature

Acknowledgements

I would like to thank doc. Ing. Martin Hromčík, Ph.D. for his counsel and for his selfless and devoted attitude towards his students.

I would also like to thank to Ing. Denis Efremov for showing me that introductions do not always have to be boring and to Ing. Marek László for his stabilizing feedback on my twin-track model.

I would like to thank to my supervisor Ing. Tomáš Haniš, Ph.D. for his valuable inputs regarding the Koopman theory.

I would like to thank to Ing. Milan Korda, Ph.D. for inviting me to Toulouse for a short-term internship and for his help with the Koopman operator framework.

Mainly I would like to thank my mother for her everlasting love and support. Without her, the life I have now would not be possible.

Declaration

I declare that the presented work was developed independently and that I have listed all sources of information used within it in accordance with the methodical instructions for observing the ethical principles in the preparation of university theses.

Prague, date 24. May 2019

.....
signature

Abstract

This work presents a vehicle dynamics control algorithm based on a novel approach in nonlinear control, which uses the Koopman operator to construct a linear predictor of a nonlinear system. The predictor is able to approximate the dynamics of the nonlinear system in a whole *subspace* of the nonlinear state-space, unlike the common linearization method based on the Taylor series, which approximates the nonlinear system only around an operating point. The procedure to construct this predictor is completely data-driven and uses only convex optimization methods. This work presents a set of nonlinear vehicle dynamics models of varying fidelity to be used for the predictor construction and validation. The contruction of the predictor is demonstrated in detail on a nonlinear singletrack model. The impact of all parameters influencing the predictor construction is discussed in detail. The linear predictor is then used in the Koopman model predictive control framework of [KM18a] to create a *nonlinear* control algorithm based entirely on *linear* model predictive control methods.

Keywords: Koopman operator, nonlinear control, model predictive control, vehicle dynamics control, vehicle models, identification

Supervisor: Ing. Tomáš Haniš, Ph.D.

Abstrakt

Tato práce představuje algoritmus pro řízení dynamiky vozidla, založený na nové metodě v nelineárním řízení, která využívá Koopmanův operátor k sestrojení lineárního prediktoru, schopného predikovat chování nelineárního systému. Prediktor je schopen aproximovat dynamiku nelineárního systému v celém *podprostoru* nelineárního stavového prostoru, narozdíl od známé metody linearizace založené na Taylorově rozvoji, která aproximuje nelineární systém pouze v okolí provozního bodu. Postup k sestrojení tohoto prediktoru je kompletně založený na datech a využívá výhradně konvexních optimalizačních metod. Tato práce představuje sadu sadu nelineárních modelů dynamiky vozidla různých přesností, která bude použita k sestrojení a validaci prediktoru. Sestrojení prediktoru je detailně demonstrováno na nelineárním dvoukolovém modelu. Dopad všech parametrů ovlivňujících sestrojení prediktoru je do detailu prodiskutován. Lineární prediktor je pak použitý v přístupu Koopmanova prediktivního řízení z [KM18a] pro vytvoření *nelineárního* řídícího algoritmu založeného výhradně na *lineárních* metodách prediktivního řízení.

Klíčová slova: Koopmanův operátor, nelineární řízení, prediktivní řízení, řízení dynamiky vozidla, modely vozidel, identifikace

Překlad názvu: Řídicí systém dynamiky vozu založený na MPC algoritmech

Contents

Figures

Tables

Chapter 1

Introduction

It is said that God created a safe paradise called Eden and created Adam and Eve to live there under the condition that they will not eat from the Tree of Knowledge. They did. God banished them from Paradise, from the safe haven, into to horrors of chaos and violence. Utilizing the gift of Knowledge, humanity has advanced quite a lot since that time. Eating from the Tree made man self-conscious, aware of his own being. It allowed him to exploit his surroundings in order to survive. It made him able to learn how the world works. To discover the natural laws binding our existence. This gave rise to physics, which can be dated back to 7^{th} century BC to philosopher Thales of Miletus, and mathematics whose roots go back to 2000 BC to ancient Mesopotamia and Egypt. This work falls into a field called control systems engineering. The study of control systems is relatively young compared to physics and mathematics, from which this field emerged. The first formal study of control systems began in 1868 with the work *On governors* [gov68] by James Clerk Maxwell who demonstrated the usefulness of mathematical models for control purposes. The field of control engineering underwent substantial changes since then. Through control of mechanical systems and linear control of SISO (Single-input single-output) systems, the field advanced quite a lot in areas such as aerospace and process control with the control of nonlinear, MIMO (Multiple-input multiple-output) systems. Today, with the everlasting growth of the electronics in our everyday lives, control engineering has seen applications in the automotive industry as well. More so with the rather recent increase in popularity of all-electric vehicles.

This gave rise to the study of vehicle dynamics from the control engineering point of view. Until then, vehicle control and stability were concerns mostly for mechanical engineers.

Most of the studies are concerned with keeping the vehicle stable and not reaching the dangerous nonlinear states. But as with everything, people push

the boundaries. The control of aircrafts did not stop when the planes were able to transport passengers, dive bomb or shoot other planes. Nowadays a fighter aircraft can perform incredible maneuvers such as the Pugachev's Cobra.
Seeing these accomplishments, one has to ask: Was is really that bad to eat the forbidden apple from the Tree of Knowledge? I think not.
Now it is time for automotive industry to incorporate advanced control system into vehicles, to make the vehicles controllable even in conditions where now only the most experienced drivers would prevail, thus making them safer. Let us have a taste from the Tree of Knowledge once again and explore the limits of vehicle handling.

This thesis is a beginning of a long journey that aims to do just that.

1.1 Goals

The aim of this work is to create a nonlinear control algorithm for control of vehicle dynamics. Vehicle features lot of nonlinearities such as coordinate systems transformations and tire models. Today, nonlinear control is done either by linearizing the model in multiple operating points or by numerically solving the nonlinear equations. Both approaches have their pros and cons. The first approach enjoys the linearity of individual systems but needs detailed knowledge of the controlled system, to know when to switch between the linearized models and how densely should the operating points be placed in the state-space. The numerical approach suffers from the computational burden of nonlinear equations while being able to provide globally optimal solutions.

A new approach has been evolving lately - the Koopman operator. The idea evolves around global linearization of the nonlinear model with a higher order linear model. Allowing the usage of the classic linear control theory to control nonlinear system in its whole state-space. The Koopman operator could be regarded as a step towards unifying the two previously mentioned approaches into one framework while using the best of both worlds: enjoying the global validity of the designed control law while exploiting the well-developed linear control theory.

This thesis uses the Koopman operator approach to represent the nonlinear vehicle model in a predefined subspace with a linear system and to design a linear control algorithm based on the linear system. The nonlinear vehicle

model will then be controlled by a linear control algorithm that is valid in the whole (operating) subspace of the nonlinear model, unlike the local linearization methods which are valid only around the operating point.

The goals of this thesis are following:

1. Derive and implement nonlinear control design and validation vehicle models.
2. Create a linear representation of the control design model in a predefined operating subspace.
3. Design model predictive control algorithm based on the linear representation of a vehicle.
4. Validate the control law on a high fidelity validation model.

1.2 Structure

This work is structured as follows:

Chapter 2	Derivation of nonlinear vehicle models for control design and validation.
Chapter 3	Explanation of the Koopman operator framework for control purposes.
Chapter 4	Linearization of the nonlinear vehicle control design model in an operating subspace.
Chapter 5	Design of a linear Model predictive control algorithm.
Chapter 6	Validation the designed algorithm on a high fidelity validation model.

Chapter 2

Mathematical vehicle models

The

All vehicle models necessary for this work will be described in this chapter. The baseline model is the 16-state nonlinear twin-track model with Pacejka tire model, all other models are derived from the baseline model.

2.1 Twin-track model

The twin-track model was based on community recognized model derivation introduces by Schramm ([SHB14]). Most of the model is implemented using MATLAB functions, where suited the Simulink environment was exploited. The purpose of this model is validation of control system design introduced in chapter Chapter 5. This model does not include suspension kinematics, meaning that the model cannot simulate various types of suspension (MacPherson strut, double wishbone, trailing-arm, etc.) and their effects (such as roll centers axis, camber change rates, etc.). The reasons for not including suspension geometry is high complexity and low payoff because this model was not designed to simulate a racecar but a generic vehicle having some basic physical parameters, which is sufficient for validation of the control algorithm. For more information about suspension geometry, see [MM96].

Linear approximation for purposes of control system design is derived based

on trimming and local equilibria linearization. Trimming tools which were also developed in MATLAB can find equilibria points for a straight-driving vehicle or for a steady-state turn.

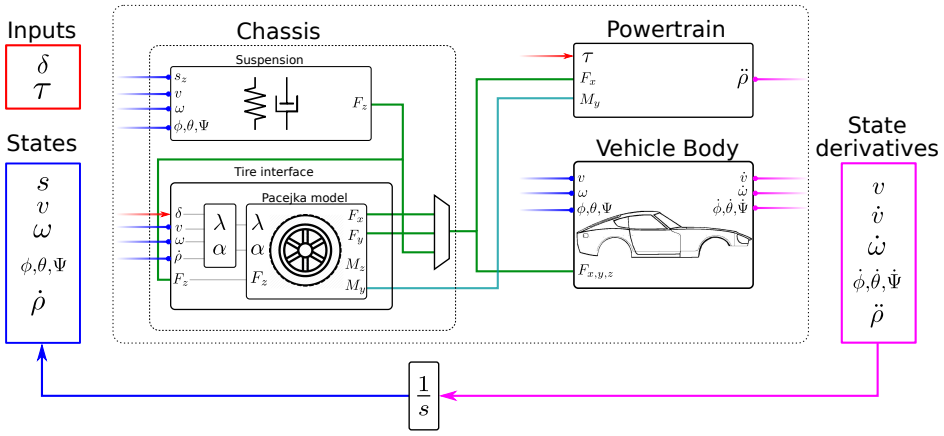


Figure 2.1: Scheme of the twin-track model.

Vehicle dynamics structure. The vehicle dynamics model is traditionally divided into 3 main parts: Chassis (suspension and tire interface), Powertrain and Vehicle Body, as seen in Fig. 2.1.

The inputs to the model are steering angles on all 4 wheels and input torques on all 4 wheels. The steering angles go directly into the Tire interface which generates the longitudinal and lateral forces acting on the Vehicle body. The input torques go to the Powertrain, where they influence the angular velocity of the wheels.

Notation. The notation used in this chapter is shown in Fig. 2.2.

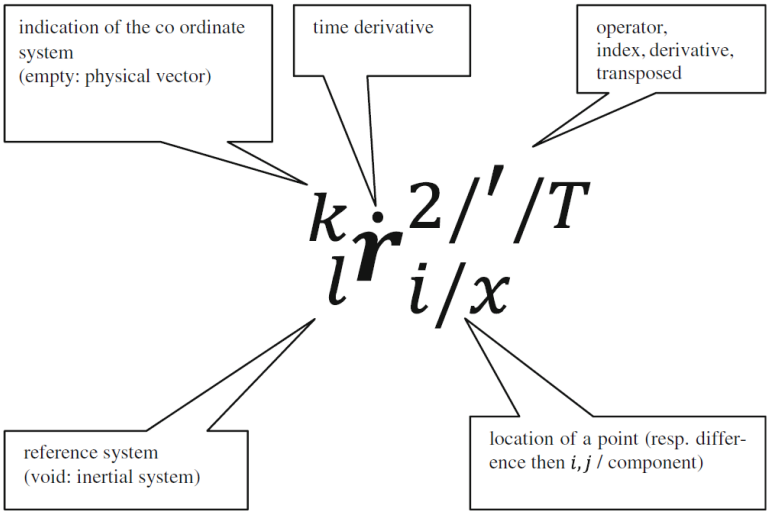


Figure 2.2: Description of the notation used in this chapter. Adopted from [SHB14] page *xvi*

State vector. The model has 16 states.

States	Units	Description	Dimensions
$\mathbf{s}_E$	(m)	Position of the vehicle body in earth-fixed frame.	$[3 \times 1]$
$\mathbf{v}_V$	(m/s)	Velocity of the vehicle body in body-fixed frame.	$[3 \times 1]$
$\boldsymbol{\omega}_V$	(rad/s)	Angular velocities the of the vehicle body in the body-fixed frame.	$[3 \times 1]$
ϕ, Θ, Ψ	(rad)	Euler angles (earth-fixed frame).	$[3 \times 1]$
$\dot{\boldsymbol{\rho}}_{R_i}$	(rad/s)	Wheel angular velocities, each in i^{th} wheel frame.	$[4 \times 1]$

The 4 wheels are numbered according to the drawing in Fig. 2.3.

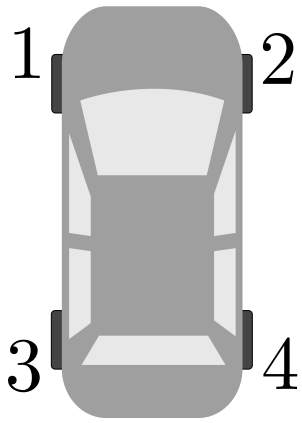


Figure 2.3: Wheel numbering

Coordinate systems. The vehicle coordinate system and inertial coordinate system are depicted in Fig. 2.4. The wheel coordinate system is depicted in Fig. 2.5.

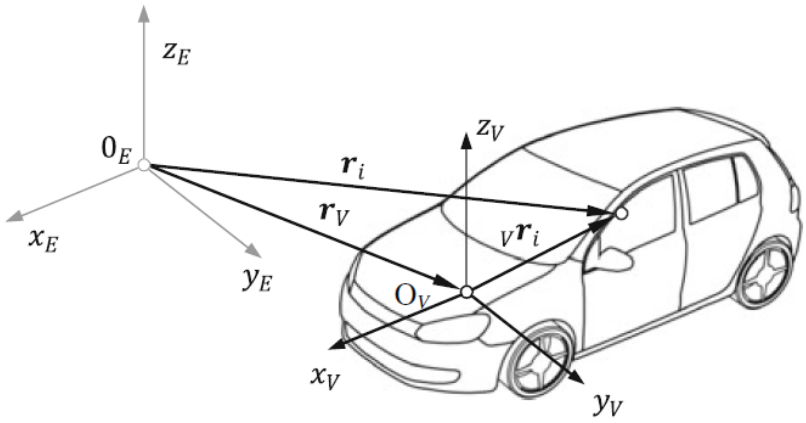


Figure 2.4: Inertial (earth-fixed) coordinate system and Vehicle (body-fixed) coordinate system.
Adopted from [SHB14]

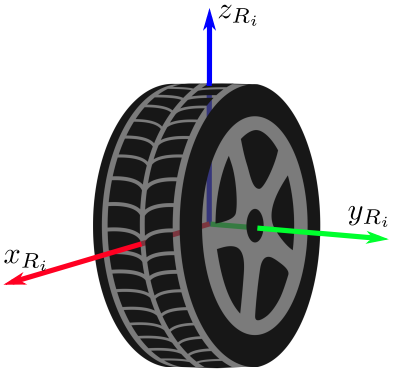


Figure 2.5: Coordinate system of i^{th} wheel.

Coordinate systems transformations. The following rotation matrices are used for transformation between the three coordinate systems. Goniometric functions *sin* and *cos* have been replaced by s and c for simplicity. Wheel-fixed coordinate system to body-fixed coordinate system can be transformed by

$$
{}^{V}\mathbf{T}_{R_i} =
\begin{bmatrix}
c(\delta_i)\, c(\Theta) & -s(\delta_i)\, c(\Theta) & -s(\Theta) \\
s(\phi)\, s(\Theta)\, c(\delta_i) + c(\phi)\, s(\delta_i) & -s(\phi)\, s(\Theta)\, s(\delta_i) + c(\phi)\, c(\delta_i) & s(\phi)\, c(\Theta) \\
c(\phi)\, s(\Theta)\, c(\delta_i) - s(\phi)\, s(\delta_i) & -c(\phi)\, s(\Theta)\, s(\delta_i) - s(\phi)\, c(\delta_i) & c(\phi)\, c(\Theta)
\end{bmatrix},
\tag{2.1}
$$

where δ_i is the steering angle of wheel i, ϕ and Θ are Euler angles (roll and pitch angle respectively).

Transformation from body-fixed frame to inertial frame is done by

$$
{}^{E}\mathbf{T}_{V} = \begin{bmatrix} c(\Theta)\,c(\Psi) & s(\phi)\,s(\Theta)\,c(\Psi) - c(\phi)\,s(\Psi) & c(\phi)\,s(\Theta)\,c(\Psi) + s(\phi)\,s(\Psi) \\ c(\Theta)\,s(\Psi) & s(\phi)\,s(\Theta)\,s(\Psi) + c(\phi)\,c(\Psi) & c(\phi)\,s(\Theta)\,s(\Psi) - s(\phi)\,c(\Psi) \\ -s(\Theta) & s(\phi)\,c(\Theta) & c(\phi)\,c(\Theta) \end{bmatrix}, \tag{2.2}
$$

where ϕ, Θ and Ψ are Euler angles: roll, pitch and roll respectively.

Inverse transformations can be obtained by transposition, since both matrices are rotation matrices.

$$
{}^{R_i}\mathbf{T}_{V} = \left({}^{V}\mathbf{T}_{R_i}\right)^{-1} = \left({}^{V}\mathbf{T}_{R_i}\right)^{T} \tag{2.3}
$$

$$
{}^{E}\mathbf{T}_{V} = \left({}^{V}\mathbf{T}_{E}\right)^{-1} = \left({}^{V}\mathbf{T}_{E}\right)^{T} \tag{2.4}
$$

For more clarification on how these matrices were derived, see [SHB14].

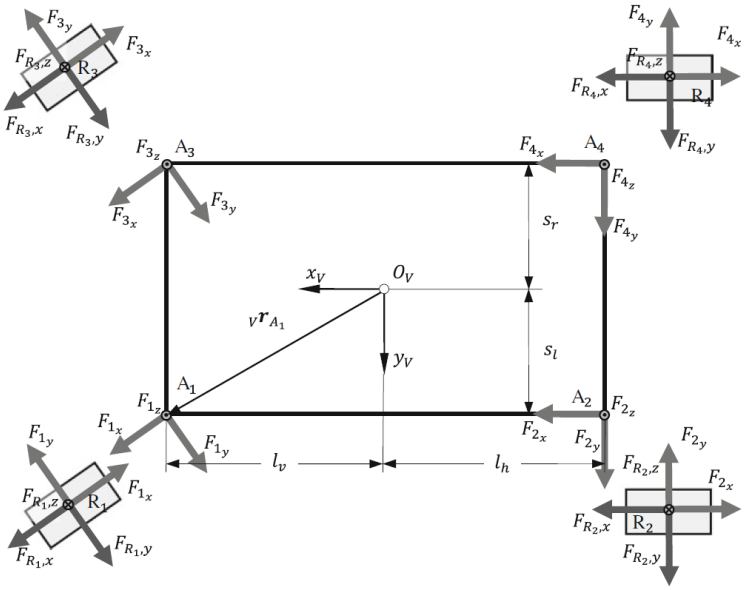


Figure 2.6: Top view of the vehicle model.
Adopted from [SHB14].

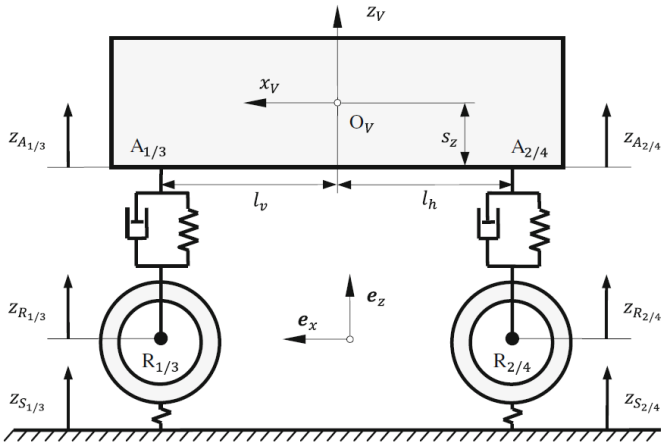


Figure 2.7: Side view of the vehicle model.
Adopted from [SHB14].

Vehicle model diagrams. Note that the tire forces calculated from tire model are denoted as $F_{R_i,x/y}$ whereas the actual force acting on the chassis is denoted as $F_{i_{x/y}}$. These forces are identical, their only difference is the coordinate system in which they are represented (wheel-fixed frame for $F_{R_i,x/y}$ and body-fixed frame for $F_{i_{x/y}}$).

2.1.1 Vehicle body

The vehicle body is modeled as a rigid body with Newton-Euler equations. The Newton equation for body-fixed frame in vector frame can be written as

$$m\left(\dot{\mathbf{v}}_V + \boldsymbol{\omega}_V \times \mathbf{v}_V\right) = \mathbf{F}, \tag{2.5}$$

where $\mathbf{F}$ denotes the vector of the total force acting on the body. All vectors are in body-fixed frame. The equation (2.5) can be expanded as

$$m_v \left(\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{v}_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \right) = \sum_{i=1}^{4} \begin{bmatrix} \mathbf{F}_{i,x} \\ \mathbf{F}_{i,y} \\ \mathbf{F}_{i,z} \end{bmatrix} - \frac{1}{2} c_w \rho A \sqrt{v_x^2 + v_y^2} \begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix} + {}^V\mathbf{T}_E \begin{bmatrix} 0 \\ 0 \\ -m_v g \end{bmatrix}. \tag{2.6}$$

Forces $\mathbf{F}_{i,x/y/z}$ are in body-fixed coordinates, the matrix ${}^V\mathbf{T}_E$ transforms the earth-fixed gravitational acceleration to body-fixed coordinates. The rest of the variables are in body-fixed frame. The term $-\frac{1}{2} c_w \rho A \sqrt{v_x^2 + v_y^2} \begin{bmatrix} v_x \\ v_y \\ 0 \end{bmatrix}$ models air-resistance, where c_w is drag coefficient, $\rho\,(kg \cdot m^{-3})$ is air density and $A\,(m^2)$ is the total surface exposed to the airflow.

The Euler equation in vector form can be written as

$$\Theta_V \dot{\boldsymbol{\omega}}_V + \boldsymbol{\omega}_V \times (\Theta \boldsymbol{\omega}_V) = \mathbf{M}, \tag{2.7}$$

where Θ_V is inertia matrix of the vehicle body and $\mathbf{M}$ is total torque acting on the body. The (2.7) can be expanded as

$$\Theta_v \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} + \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \times \left(\Theta_v \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \right) = \sum_{i=1}^{4} \mathbf{r}_i \times \begin{bmatrix} \mathbf{F}_{i,x} \\ \mathbf{F}_{i,y} \\ \mathbf{F}_{i,z} \end{bmatrix} + \mathbf{r}_w \times \mathbf{F}_w, \tag{2.8}$$

where $\mathbf{F}_w$ is a vector of aerodynamic forces acting at point $\mathbf{r}_w$, the vector $\mathbf{r}_w$ is w.r.t. center of gravity in vehicle coordinates. It can be used to shift the center of aerodynamic pressure. The vectors $\mathbf{r_i}$ are set according to Fig. 2.6 and Fig. 2.7.

$$\mathbf{r} = \begin{bmatrix} \mathbf{r_1} \\ \mathbf{r_2} \\ \mathbf{r_3} \\ \mathbf{r_4} \end{bmatrix}^T = \left[\begin{bmatrix} l_v \\ s_l \\ -s_z \end{bmatrix}, \begin{bmatrix} l_v \\ -s_r \\ -s_z \end{bmatrix}, \begin{bmatrix} -l_h \\ s_l \\ -s_z \end{bmatrix}, \begin{bmatrix} -l_h \\ -s_r \\ -s_z \end{bmatrix} \right] \tag{2.9}$$

2.1.2 Chassis

Suspension

The suspension is modeled as spring-damper systems acting on each wheel individually.

Spring force acting on i^{th} wheel is defined as follows

$${}^V\mathbf{F}_{Fi} = -\left(c_{ai} \Delta l_{Fi}\right) {}^V\mathbf{T}_E \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \forall i \in \{1, 2, 3, 4\}, \tag{2.10}$$

where $c_{ai}\ (N/kg)$ is the stiffness coefficient of spring i. $\Delta l_{F_i}\ (m)$ is the compression of spring i, ${}^V\mathbf{T}_E$ is a rotation matrix, transforming inertial coordinates to vehicle coordinates. The multiplication by the vector $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ means that the force acts only along the (inertial) z_E-axis (the spring is assumed to always

point upwards with respect to the inertial coordinates). Damping force acting on i^{th} wheel is defined as follows

$$ {}^{V}\mathbf{F}_{Di} = -\left(d_{ai}\Delta\dot{l}_{Fi}\right) {}^{V}\mathbf{T}_{E} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \forall i \in \{1,2,3,4\} \tag{2.11} $$

Where $d_{ai}\,(N \cdot s/m)$ is the damping coefficient.

Tire interface

Slip variables used by tire models will be defined here. All tire models use slip ratio λ and slip angle α as their inputs. The notation in this section will differ from [SHB14] at some places, most importantly, the slip angle will be denoted by λ instead of s.

Slip variables. Longitudinal (circumferential) slip:

$$ \lambda_i = \frac{{}^{R_i}\dot{x}_{R_i} - r\dot{\rho}_{R_i}}{\max\left(|r\dot{\rho}_{R_i}|, |\,{}^{R_i}\dot{x}_{R_i}|\right)} \tag{2.12} $$

Slip angle:

$$ \alpha_i = -\arctan\left(\frac{{}^{R_i}\dot{y}_{R_i}}{|\,{}^{R_i}\dot{x}_{R_i}|}\right) \tag{2.13} $$

Where ${}^{R_i}\dot{x}_{R_i}/{}^{R_i}\dot{y}_{R_i}$ is velocity of the wheel center point along x/y axis in the wheel-fixed coordinate system , r is wheel radius (note that there is no distinction between effective and nominal radius of the wheel, for simplicity) and $\dot{\rho}_{R_i}$ is angular velocity of wheel i

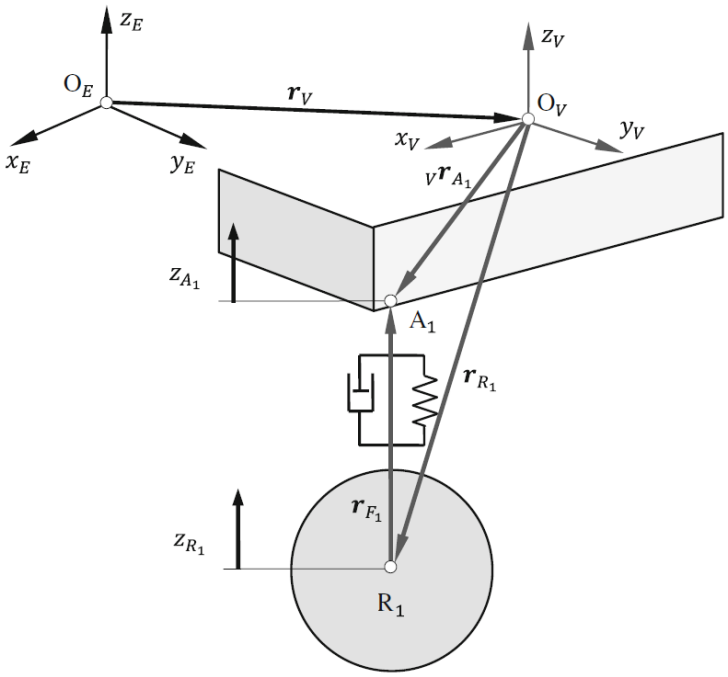


Figure 2.8: Clarification on the meaning of wheel center point vector r_{R_i} and the pivot point A_i . Both vectors are in the body-fixed frame, with respect to the origin O_V.
Adopted from [SHB14].

The wheel center point velocities can be obtained from

$$\begin{bmatrix} {}_{R_i}\dot{x}_{R_i} \\ {}_{R_i}\dot{y}_{R_i} \\ {}_{R_i}\dot{z}_{R_i} \end{bmatrix} = {}^{R_i}\mathbf{v}_{R_i} = {}^{R_i}\mathbf{T}_V \, {}^{V}\mathbf{v}_{R_i} \tag{2.14}$$

Where ${}^{V}\mathbf{v}_{R_i}$ is wheel center point velocity with respect to body-fixed coordinates and can be obtained from

$${}^{V}\mathbf{v}_{R_i} = \begin{bmatrix} {}^{V}\dot{x}_{R_i} \\ {}^{V}\dot{y}_{R_i} \\ {}^{V}\dot{z}_{R_i} \end{bmatrix} = {}^{V}\mathbf{v}_V + {}^{V}\omega_V \times {}^{V}\mathbf{r}_{R_i} + {}^{V}\mathbf{T}_E \begin{bmatrix} 0 \\ 0 \\ -\dot{l}_{F_i} \end{bmatrix}, \forall i \in \{1,2,3,4\} \tag{2.15}$$

Where ${}^{V}\mathbf{v}_V$ is vehicle velocity with respect to vehicle coordinate system, ${}^{V}\omega_V$ is vehicle angular velocity with respect to vehicle coordinate system and ${}^{V}\mathbf{r}_{R_i}$ is the position of the wheel center point with respect to vehicle coordinate system and can be calculated as

$${}^{V}\mathbf{r}_{R_i} = {}^{V}\mathbf{r}_{A_i} + {}^{V}\mathbf{T}_E \begin{bmatrix} 0 \\ 0 \\ -l_{F_i} \end{bmatrix}, \forall i \in \{1,2,3,4\}, \tag{2.16}$$

where ${}^{V}\mathbf{r}_{A_i}$ is position of the spring anchor with respect to the vehicle coordinate frame and l_{F_i} is length of the spring.

The vector ${}^{V}\mathbf{r}_{A_i}$ points to the point where the spring is anchored to the vehicle chassis and where the tire forces are applied to the chassis as seen in Fig. 2.6 and Section 2.1.2.

2.1.3 Tire models

Three tire models with varying levels complexity and accuracy will be described now. Simplified Pacejka model, simplified Pacejka with friction ellipse and full Pacejka tire model.

Simplified Pacejka. The simplified model was modelled according to [Lor12]. The model uses constant coefficients B, C, D, E for the Magic formula:

$$F = D\cos\left(C\arctan\left(Bx - E\left(Bx - \arctan\left(Bx\right)\right)\right)\right) \tag{2.17}$$

where x is either sideslip angle α or longitudinal slip λ. F is either F_y,M_z or F_x, depending on the input argument x. For calculating the longitudinal force F_x, the argument x would be substituted by λ whereas F_y and M_z are calculated by substituting α. Coefficients B, C, D, E are generally time-variant and dependant on what F means. In this model, B, C, D, E are constant for given F. So for calculating F_y,M_z and F_x, one would need 3 sets of these parameters.

This approach makes the forces F_x and F_y independent on each other, which is never the case in the real world. This dependency is often expressed with traction ellipse (also called friction ellipse or Kamm's circle).

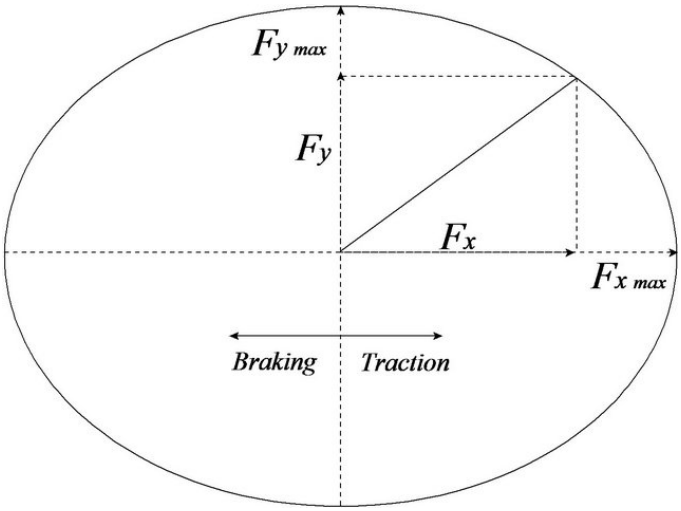


Figure 2.9: Traction ellipse for constant parameters B, C, D and E.
x-axis is for λ, y-axis is for α. Note that maximum force can be achieved only when either λ or α are zero.
Adpoted from [Lib16].

A method called "Combined Slip with friction ellipse" adopted from a document describing the tire model in *Adams* multibody simulator ([Lor12]) is used to capture this dependency.

Combined Slip with friction ellipse. Let's call the forces calculated from (2.17) $F_{x,0}$ and $F_{y,0}$. The slip ratio λ and slip angle α are used to calculate the following:

$$\alpha^* = \sin(\alpha) \tag{2.18}$$

$$\beta = \arccos\left(\frac{|\lambda|}{\sqrt{\lambda^2 + \alpha^{*2}}}\right). \tag{2.19}$$

Then friction coefficients are defined.

$$\mu_{x,act} = \frac{F_{x,0}}{F_z} \qquad \mu_{y,act} = \frac{F_{y,0}}{F_z} \tag{2.20}$$

$$\mu_{x,max} = \frac{D_x}{F_z} \qquad \mu_{y,max} = \frac{D_y}{F_z} \tag{2.21}$$

$$\mu_x = \frac{1}{\sqrt{\left(\frac{1}{\mu_{x,act}}\right)^2 + \left(\frac{\tan(\beta)}{\mu_{y,max}}\right)^2}} \tag{2.22}$$

$$\mu_y = \frac{\tan(\beta)}{\sqrt{\left(\frac{1}{\mu_{x,max}}\right)^2 + \left(\frac{\tan(\beta)}{\mu_{y,act}}\right)^2}} \tag{2.23}$$

Finally, the forces from (2.17) are scaled using the friction coefficients.

$$F_x = \frac{\mu_x}{\mu_{x,act}} F_{x,0} \tag{2.24}$$

$$F_y = \frac{\mu_y}{\mu_{y,act}} F_{y,0} \tag{2.25}$$

Forces F_x and F_y are now respecting the traction ellipse from Fig. 2.9. The force F_z is the tire load, the resultant force from the spring-damper system.

Pacejka2002. The last considered model is Pacejka2002, implemented according to well-known work of Hans B. Pacejka ([Pac02]), which is the current industry standard for tire simulation.

This model is very accurate, unfortunately parametrizing it is rather difficult. The model uses over 100 parameters that cannot be derived from physical properties of the tire but instead they have to be measured. Also the Pacejka tire model changed a little with each release of [Pac02] (see [Pac12] for example) so even after finding a publicly available set of parameters one has to verify for which version of the model were they measured.

This work uses the model Pacejka2002 as introduced in [Pac02]. The parameters were adopted from the Automotive challenge 2018 [Aut18] organized by Rimac Automobili.

2.1.4 Powetrain

Rather simple model of Powertrain is considered. For wheel i, it holds that

$$J_{R_i} \ddot{\rho}_{R_i} = M_{a,R_i} - M_{b,R_i} sign\left(\dot{\rho}_{R_i}\right) - r F_{R_i,x}, \tag{2.26}$$

where J_{R_i} is the wheel moment of inertia, $\dot{\rho}_{R_i}$ is the angular velocity of the wheel. M_a and M_b are input and braking torques respectively, r is the wheel radius and $F_{R_i,x}$ is the longitudinal force of the wheel (in wheel-fixed coordinate system).

Enhancement is done by limiting the input torque M_a by a constant maximum torque $T_{max}\,(N \cdot m)$ or by the motor maximum power $P_{max}\,(kW)$, according to the equation (2.27).

$$M_{a,max} = \frac{P_{max}}{\dot{\rho}_{R_i}}, \tag{2.27}$$

Comparison between $M_{a,max}$ and T_{max} can be seen in Fig. 2.10.

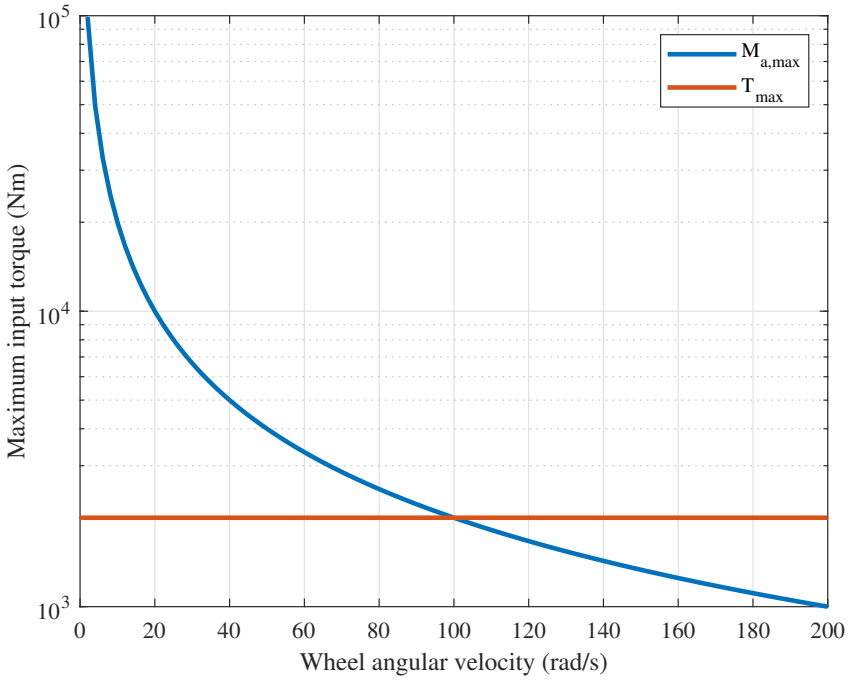


Figure 2.10: Blue line represents the variable limit on M_a influences by the motor maximum power and the current wheel angular velocity. Red line is a constant maximum torque limitation. For each wheel angular velocity $\dot{\rho}_{R_i}$, the lesser of the two values is used to saturate the input torque M_a. The plotted values are for $P_{max} = 250kW$ and $T_{max} = 2000Nm$.

2.2 Single-track model

This section describes single-track model, which will be used for control design in Chapter 5. The single-track is a planar model with its four wheels merged into two. The model can have 5 or 3 states, depending on whether the Powertrain dynamics is present (Section 2.2.1) or not (Section 2.2.2).

The following subsections will present 3 versions of the single-track model, all derived from the twin-track model Section 2.1.

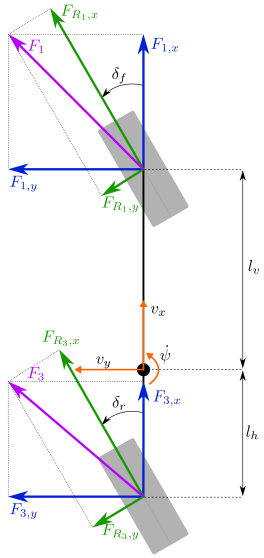


Figure 2.11: The single-track model. Forces F_{R_2} and F_{R_4} are not depicted in the figure because in a general case with symmetric tires $F_{R_2} = F_{R_1}$ and $F_{R_4} = F_{R_3}$.

2.2.1 5 state single-track

The single-track model includes the Powertrain and is depicted in Fig.Fig. 2.11. The states and parameters of this model are a subset of states and parameters of Section 2.1. State vector of the single-track model is

$$\mathbf{x_{single}} = \begin{bmatrix} v_x\,(m/s) \\ v_y\,(m/s) \\ \dot{\psi}\,(rad/s) \\ \dot{\rho}_f\,(rad/s) \\ \dot{\rho}_r\,(rad/s) \end{bmatrix} \tag{2.28}$$

where $v_{x/y}$ is longitudinal/lateral velocity in body-fixed coordinates, $\dot{\psi}$ is yawrate and $\dot{\rho}_{f/r}$ are front/rear wheel angular rates.

This model has 4 wheels, with two wheels always being in the same place. This allows for usage of asymmetric tire models, reduces the work needed to transition to twin-track model and is less error-prone than the standard approach (with only two tires modeled, one has to remember that each tire should generate twice as much force).

The vehicle body is modeled as a rigid body using Newton-Euler equations, similarly as in Section 2.1.1.

$$m_v \left(\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \end{bmatrix} + \dot{\psi} \begin{bmatrix} -v_y \\ v_x \end{bmatrix} \right) = \sum_{i=1}^{4} \begin{bmatrix} \mathbf{F}_{i,x} \\ \mathbf{F}_{i,y} \end{bmatrix} - \frac{1}{2} c_w \rho A_w \sqrt{v_x^2 + v_y^2} \begin{bmatrix} v_x \\ v_y \end{bmatrix} \tag{2.29}$$

$$J_{zz} \ddot{\psi} = \sum_{i=1}^{4} \mathbf{r}_i \mathbf{F}_{i,y} \tag{2.30}$$

$$J_{R_i} \ddot{\rho}_{R_i} = M_{a,R_i} - M_{b,R_i} sign\left(\dot{\rho}_{R_i}\right) - r F_{R_i,x}, \forall i = 1,3 \tag{2.31}$$

Where

$$\mathbf{r} = \begin{bmatrix} \mathbf{r_1} \\ \mathbf{r_2} \\ \mathbf{r_3} \\ \mathbf{r_4} \end{bmatrix}^T = \left[\begin{bmatrix} l_v \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} l_v \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} -l_h \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} -l_h \\ 0 \\ 0 \end{bmatrix} \right] \tag{2.32}$$

is the vector describing position of each wheel with respect to the center of gravity. The wheels are numbered in this order: front-left, front-right, rear-left, rear-right. m_v is the vehicle mass, $\mathbf{F}_{i,x/y}$ is a force acting on i-th wheel along x/y axis in body-fixed coordinates. $F_{R_i,x}$ is a force acting along x axis in wheel coordinate system (direct output of the tire model). The term $-\frac{1}{2} c_w \rho A_w \sqrt{v_x^2 + v_y^2} \begin{bmatrix} v_x \\ v_y \end{bmatrix}$ is an approximation of air-resistance, c_w is a drag coefficient, ρ is air density and A_w is the total surface exposed to the air flow. J_{zz} is the vehicle inertia about z-axis. J_{R_i} the wheel inertia about y-axis. Inputs are wheel torques M_{a,R_i} (throttle), M_{b,R_i} (break) and steering angles $\delta_{f/r}$.

Since this is a 4-wheel single-track model, the following holds:

$$\dot{\rho}_{R_1} = \dot{\rho}_{R_2} \tag{2.33}$$

$$\dot{\rho}_{R_3} = \dot{\rho}_{R_4}. \tag{2.34}$$

The forces $\begin{bmatrix} F_{R_i,x} \\ F_{R_i,y} \end{bmatrix}$ are the output of the tire model from Section 2.1.2.

2.2.2 3 state single-track

To further simplify the model, (2.31) can be omitted, resulting in a model with only 3 states: $\begin{bmatrix} v_x & v_y & \dot{\psi} \end{bmatrix}^T$. The inputs are then longitudinal slips

(which were previously derived from ((2.31))) and steering angles. Note that this assumes existence of a longitudinal slip controller (traction control).

2.2.3 3 state single-track without tire model

To simplify the model even more, one can omit the tire model and use the tire forces $\begin{bmatrix} F_{i,x} \\ F_{i,y} \end{bmatrix}$ as input, assuming the existence of a higher level control system controlling the tire forces and thus securing the assumption that the car can be controlled directly by force reference.

Chapter 3

The Koopman operator

Koopman operator for representing nonlinear dynamics of a vehicle by a higher-order linear system is used in this work. The linear representation of vehicle dynamics will then be used for control system design. The vehicle system typically features a lot of nonlinearities such as rigid-body dynamics, coordinate system transformations and most importantly the tire model using Pacejka magic formula. The aim of this thesis is to represent these nonlinearities by a higher-order linear system obtained from the Koopman operator framework. The nonlinear system will then be controlled by a linear controller based on the linear Koopman system, allowing to control the nonlinear system in a predefined subspace of the nonlinear state-space using well-developed linear control theory. Here, the method of choice is linear MPC but any other linear control methods such as LQR and H_∞ (see [SS05]) can be used.

3.1 Basic idea of the Koopman operator

The theory of the Koopman operator states that any nonlinear uncontrolled dynamical system can be globally represented by a linear system of infinite order with zero error. This idea was first introduced by Koopman in [Koo31]. The idea to use the Koopman operator for control was proposed by Mezić ([MB04]) in his work dealing with exploitation of the Koopman operator to compare dynamics of physical systems and their nonlinear models. The extension of the Koopman operator framework to controlled nonlinear dynamical systems was done very recently by Korda in [KM18a], where the framework

for designing a linear MPC based on Koopman operator was first introduced.

The Koopman operator framework. Consider discrete-time nonlinear system with sampling time $T_s\,(s)$:

$$x_{k+1} = f\left(x_k, u_k\right), \tag{3.1}$$

where k denotes the sample at time $k{\cdot}T_s$, u_k is the input vector with dimension N_u and x_k is the state vector with dimension N_x.

The vector x_k is transformed by so-called *basis functions* $\phi : \mathbb{R}^{N_x} \to \mathbb{R}$ to a higher order linear state space of the Koopman operator. The state space of the Koopman operator will be denoted as *lifted* space in this work and the lifted state vector will be denoted by z_k. The lifted state space evolves linearly with the system (3.2). Note that the Koopman operator is generally infinite-dimensional, the system (3.2) (or more precisely, the matrix A) is its finite-order approximation.

$$\begin{aligned} z_{k+1} &= Az_k + Bu_k \\ y_k &= Cz_k \\ &\qquad\qquad \text{for } z_0 = \phi\left(x_0\right) \end{aligned} \tag{3.2}$$

A, B and C are complex matrices of a discrete-time LTI (linear time-invariant) system with lifted state vector z_k. The lifted state vector z_k can be transformed back to the original nonlinear space via linear transformation if the output vector y_k is selected as x_k (as is the case in this thesis), meaning that $x_k = Cz_k$. The relationship between (3.1) and (3.2) is shown in Fig. 3.1.

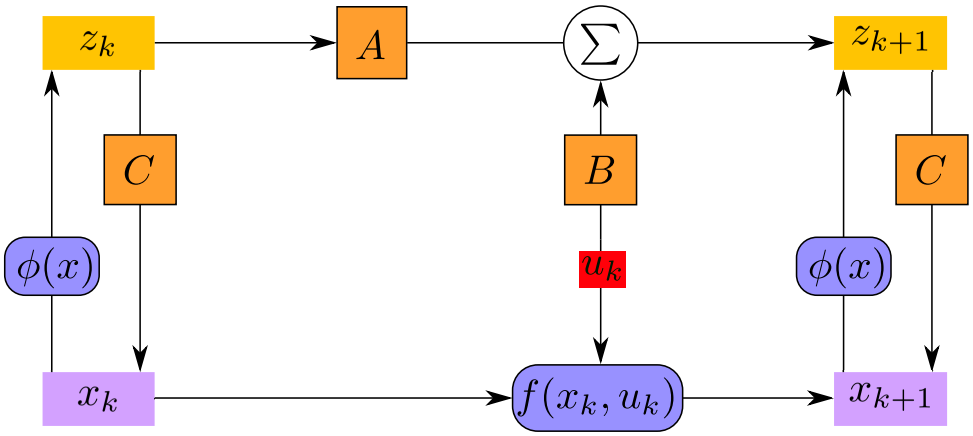


Figure 3.1: Discrete-time scheme showing the relationship of a nonlinear system $x_{k+1} = f\left(x_k, u_k\right)$ and linear Koopman system $z_{k+1} = Az_k + Bu_k$. The nonlinear state vector x_k can be lifted upwards to the linear state vector z_k. The lifted state vector z_k can be then projected back down to x_k via multiplication by the matrix C.

3.1.1 One-step prediction with general basis functions

A method to approximate the matrices A and B simultaneously from data was proposed in [KM18a]. The method uses Extended Dynamic mode decomposition (EDMD) [WKR15] to create LTI system functioning as a one-step predictor. The selection of basis functions isn't clear however and there is no deterministic way on how to select them in order to approximate the system with desired accuracy. This approach was used on a singletrack model in [CHH]. However, the approach proved to be very sensitive to the selection of the basis functions ϕ, rendering the method unsuitable for some systems, such as the singletrack model.

3.1.2 Multi-step prediction with deterministic basis function selection

Then a method providing a formal description for the basis functions ϕ was introduced in [KM18b].

Consider a basis function defined according to (3.3)

$$\phi\left(x_k^j\right) = \lambda^k g\left(x_0^j\right), \tag{3.3}$$

where $\lambda \in \mathbb{C}$, $g : \mathbb{R}^{N_x} \to \mathbb{R}$ is generally any function and x_0^j is a starting point of some trajectory j of the system (3.1) and x_k^j is a point on the *same* trajectory at sample k. The set of starting points x_0^j will be denoted as Γ.

Note that for the basis function (3.3) it holds that

$$\phi\left(x_{k+1}^j\right) = \lambda\phi\left(x_k^j\right), \tag{3.4}$$

because

$$\phi\left(x_{k+1}^j\right) = \lambda^{k+1} g\left(x_0^j\right) = \lambda \cdot \lambda^k g\left(x_0^j\right) = \lambda\phi\left(x_k^j\right). \tag{3.5}$$

This means that the function evolves *linearly*, i.e. the future value of ϕ can be predicted by multiplying the current value by λ. Such a function is defined as an *eigenfunction* of the Koopman operator in [KM18b] and the complex number λ is the associated eigenvalue. The rest of this thesis will consider only basis functions defined according to (3.3), so the terms *basis function* and *eigenfunction* will be used interchangeably.

The theory of Koopman eigenfunctions is beyond the scope of this thesis. For more explanation, please see [KM18b].

Reconstruction of A matrix. Note that in the method from Section 3.1.1, the matrix A (along with B) was constructed from the data based on provided basis functions. In this case however, the approach is reversed. The eigenfunctions are constructed from data and the matrix A is then simply a diagonal matrix with their eigenvalues on the diagonal. Equation (3.4) describes a scalar case with $A = \lambda$.

Multi-step prediction. The definition (3.3) allows to formulate an optimization problem that allows to approximate the *whole* trajectory j, i.e. all x_k^j for $k = 1, 2, ..K$, where K is the total number of samples. This contrasts with the approach from [KM18a], mentioned in Section 3.1.1, which is concerned with predicting only the next step x_{k+1}^j.

The aforementioned optimization problem is formulated in Section 3.2.1 and the capability of multi-step prediction is then exploited in Section 3.2.3.

3.2 Selection of g functions

Although the basis functions defined according to (3.3) have now a formal description, the values of the parameter λ and the function g are still unknown. This section describes the selection of g. The heuristics for choosing λ are described in Section 4.1.1.

The selection of the g functions can be done with two approaches. In the first approach, described in the first draft of [KM18b], the g functions as well as the eigenvalues were selected by the user. More accurately, the sets $\Lambda = \{\lambda_1, \lambda_2, ...\lambda_{N_\Lambda}\}$ and $G = \{g_1, g_2, ...g_{N_g}\}$ were chosen. So for some general point x in the nonlinear state space, each eigenfunction $\phi_{\lambda,g}(x)$ had *some* value dependent on the selection of G and Λ. In order to retrieve the nonlinear, *unlifted* states, the matrix C was optimized to provide optimal linear combination of the eigenfunctions that would fit the desired output (the state vector x). This approach was used on the singletrack model in [CHH] and showed much better results than the EDMD approach mentioned in Section 3.1.1.

The new approach from [KM18b] creates the g functions, or rather their boundary values in the Γ set, optimally. The boundary values of G are optimized so that the eigenfunctions immediately provide the system output. The set Λ is still selected by the user, because as of now there is no method

to select the set optimally. But the selection of g is now deterministic and the algorithm requires less input based on some knowledge about the system, making the approach more general. This approach will be described in the following subsection.

3.2.1 Finding optimal g functions

For p^{th} output on some trajectory j at time-step k, the equation (3.6) must hold:

$$y_{p,k}^{j} = \phi_{1,p}\left(x_k^j\right) + \phi_{2,p}\left(x_k^j\right) + \cdots + \phi_{N_\Lambda,p}\left(x_k^j\right), \tag{3.6}$$

where $\phi_{i,p}$ is a basis function defined according to (3.3), associated with i^{th} eigenvalue and p^{th} output. The equation (3.6) can be expanded with (3.3) as

$$y_{p,k}^{j} = \lambda_1^k g_{p,1}^j + \lambda_2^k g_{p,2}^j + \lambda_3^k g_{p,3}^j + ...\lambda_{N_\Lambda}^k g_{p,N_\Lambda}^j, \tag{3.7}$$

where $g_{p,i}^j$ is boundary value (at time $t = 0$) of the function $g_{p,i}^j(x)$ associated with p^{th} output y_p, j^{th} trajectory and i^{th} eigenvalue λ_i. In other words, $g_{p,i}^j = g_{p,i}^j(x_0)$. The values $g_{p,i}^j$ are optimized so that (3.7) holds (at least approximately), for a given set $\Lambda = \{\lambda_1, ...\lambda_{N_\Lambda}\}$ as is shown in Fig. 3.2. Evolution of (3.7) in time is shown in Fig. 3.3. Note that each trajectory has different $g_{p,i}^j$ values. So there is $N_y \times N_\Lambda$ $g_{p,i}^j$ values per trajectory and $N_y \times N_\Lambda \times N_T$ $g_{p,i}^j$ values in total, with N_T being the number of trajectories, N_Λ the number of eigenvalues and N_y the size of the output vector.

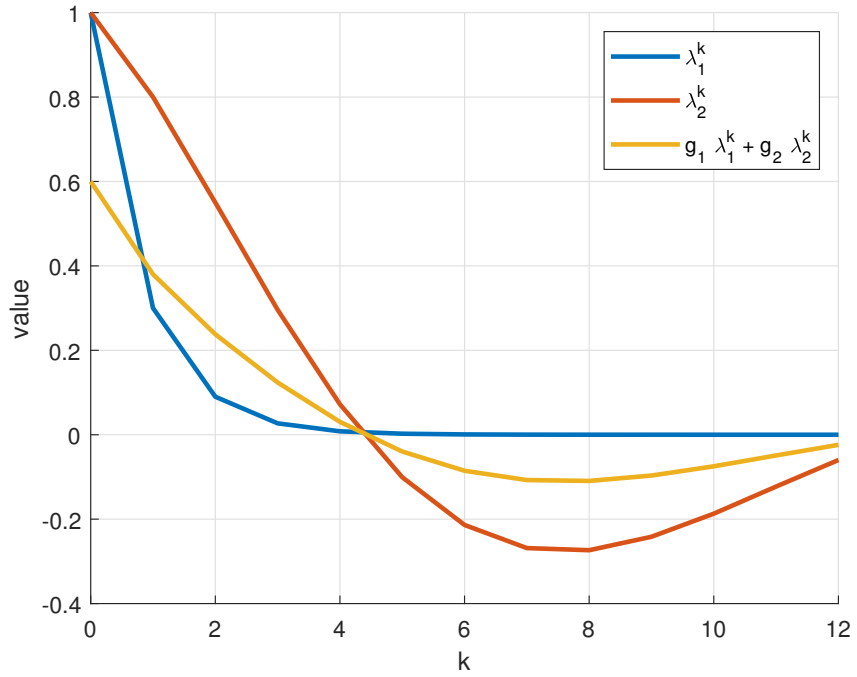


Figure 3.2: The yellow trajectory y_p is approximated by $\Re\{g_1\lambda_1^k + g_2\lambda_2^k\}$. Values used in this example are $\lambda_1 = 0.3$, $\lambda_2 = 0.8 + 0.3i$ and $(g_1, g_2) = (0.2, 0.4)$.

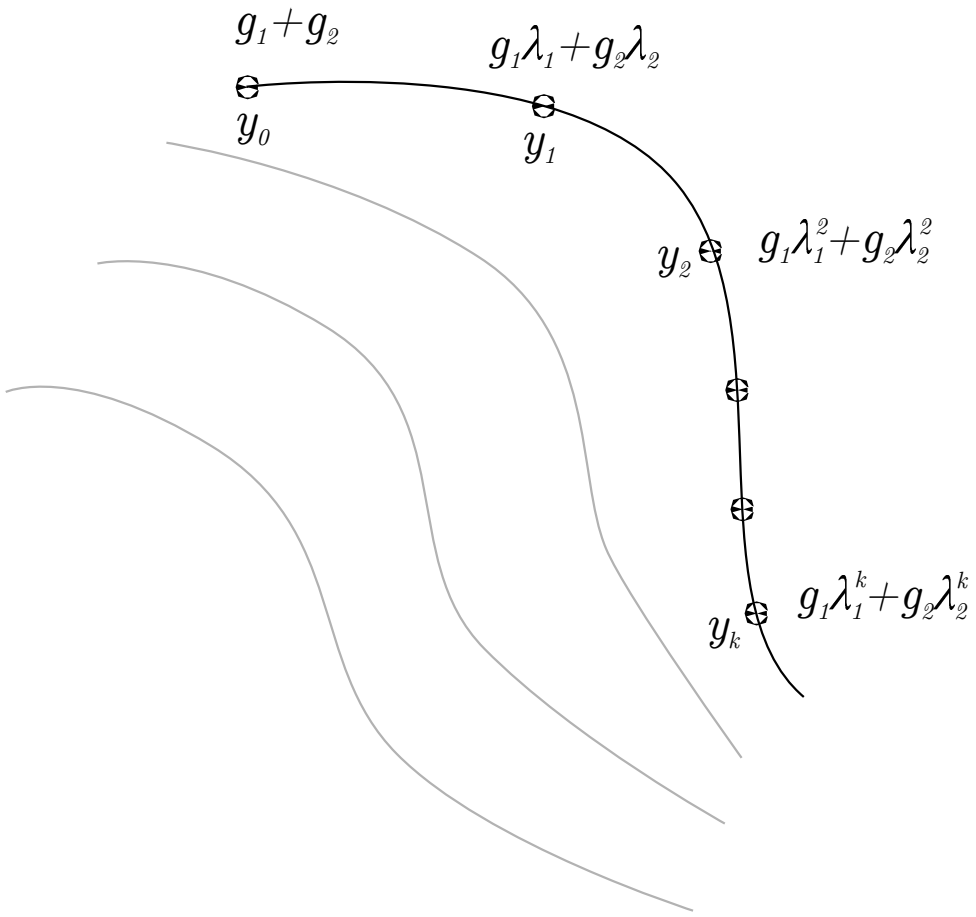


Figure 3.3: Visualization of (3.7) on a concrete trajectory for $N_\lambda = 2$. The indices p and j were dropped for simplicity.

The equation (3.7) can be approximated by a least-squares problem

$$||Lg_p - F_p||_2^2 + \gamma||DLg_p||_2^2 + \zeta||g_p||_2^2, \tag{3.8}$$

where the first term performs the approximation of (3.7) and the second and third term provide regularization controlled by the parameters γ and ζ. All three terms will be described below in detail.

Data fitting. The first term, $||Lg_p - F_p||_2^2$ is a matrix form of (3.7) for

$$L = \Big[L_1, L_2, \dots L_{N_\Lambda}\Big]_{(N_\Lambda \cdot N_T)\times(N_\Lambda \cdot N_T)}, \tag{3.9}$$

where

$$L_i = \begin{bmatrix} \begin{bmatrix} \lambda_i^1 \\ \lambda_i^2 \\ \vdots \\ \lambda_i^K \end{bmatrix}_{K\times 1} & & & \\ & \begin{bmatrix} \lambda_i^1 \\ \lambda_i^2 \\ \vdots \\ \lambda_i^K \end{bmatrix} & & \\ & & \ddots & \\ & & & \begin{bmatrix} \lambda_i^1 \\ \lambda_i^2 \\ \vdots \\ \lambda_i^K \end{bmatrix} \end{bmatrix}_{(K\cdot N_T)\times N_T} \tag{3.10}$$

with K being the number of samples in each trajectory, N_T being the number of trajectories and

$$F_p = \begin{bmatrix} \begin{bmatrix} y_{p,1} \\ y_{p,2} \\ \vdots \\ y_{p,K} \end{bmatrix}^1 \\ \begin{bmatrix} y_{p,1} \\ y_{p,2} \\ \vdots \\ y_{p,K} \end{bmatrix}^j \\ \vdots \\ \begin{bmatrix} y_{p,1} \\ y_{p,2} \\ \vdots \\ y_{p,K} \end{bmatrix}^{N_T} \end{bmatrix}_{(K\cdot N_T)\times 1} , \tag{3.11}$$

where F_p is a column vector of p^{th} outputs from all N_T trajectories (trajectories are denoted by superscript of the submatrices).

Regularization of nearest neighbours. The second term, $\gamma||DLg_p||_2^2$ is used for regularization or *smoothening* the eigenfunction values. The matrix D is constructed according to (3.12).

$$D = D_s \cdot S \tag{3.12}$$

for

$$D_s = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_{k_D \cdot K \cdot N_T} \end{bmatrix}^{-1}, \tag{3.13}$$

where D_s is a diagonal matrix of distances between two neighbouring points from the sampled trajectories and

$$S = \begin{bmatrix} \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} & & & \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}_{K \cdot N_T} \\ & \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} & & \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}_{K \cdot N_T} \\ & \ddots & & \vdots \\ & & \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}_{k_D \times 1} & \begin{bmatrix} 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 \end{bmatrix}_{K \cdot N_T} \end{bmatrix}_{(k_D \cdot K \cdot N_T) \times (K \cdot N_T)}, \tag{3.14}$$

where S is a matrix that *selects* the closest neighbours - each row selects two neighbouring points. For each point from the sampled trajectories, k_D nearest neighbours (across all trajectories) are selected by S, eigenfunction values at these k_D points are evaluated (by Lg_p) and the *difference* of these values is penalized with respect to the distance (in D_s) between the neighbouring points. The effect of regularization can be seen in Fig. 3.4.

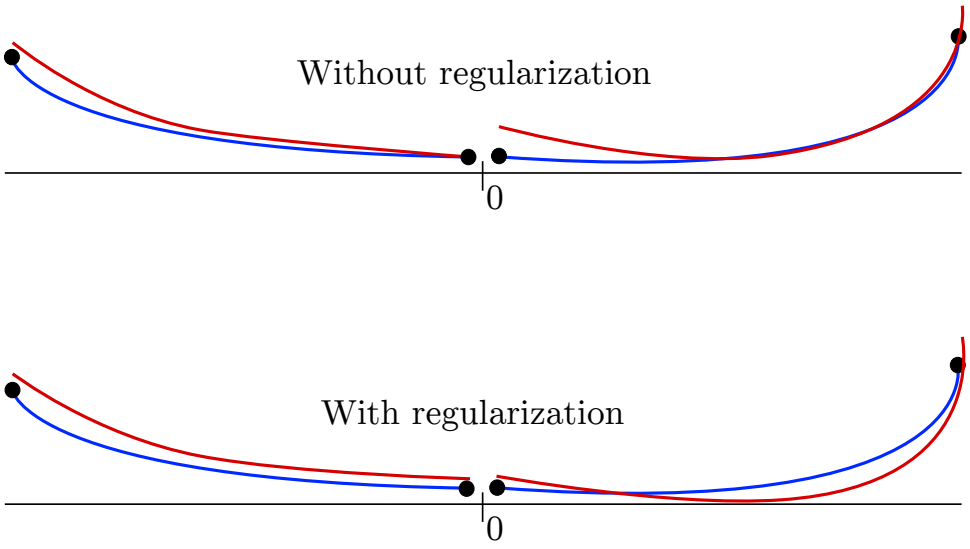


Figure 3.4: Blue lines represent output values along some trajectories, the outputs are approximated by red eigenfunctions. Without regularization, the eigenfunctions have large difference in values near the point 0. With regularization, the difference is lessened for the cost of worse fit on the individual trajectories.

Regularization of g_p. The last term $\zeta||g_p||_2^2$ is used to control the magnitude of the values in g_p. This is used to prevent numerical problems which may be caused by magnitudes close to the machine-precision.

3.2.2 Matrices A and C

Having the eigenfunctions ϕ estimated, the matrices A and C are trivial. The A matrix is a diagonal matrix with the eigenvalues from Λ on the diagonal. The Λ set is repeated N_y times, ordering depends on the order of the ϕ functions. The previous approach results in $N_y \cdot N_\lambda$ eigenfunctions. Let $\phi_{p,i}$ be an eigenfunction associated with p^{th} output and i^{th} eigenvalue. Then let

Φ be a vector containing all eigenfunctions in the following order:

$$\Phi = \begin{bmatrix} \phi_{1,1} \\ \phi_{1,2} \\ \vdots \\ \phi_{1,N_\Lambda} \\ \phi_{2,1} \\ \phi_{2,2} \\ \vdots \\ \phi_{2,N_\Lambda} \\ \vdots \\ \phi_{N_y,1} \\ \phi_{N_y,2} \\ \vdots \\ \phi_{N_y,N_\Lambda} \end{bmatrix}. \tag{3.15}$$

Then

$$A = \begin{bmatrix} \lambda_1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & \lambda_{N_\Lambda} & & & & & & \\ & & & \lambda_1 & & & & & \\ & & & & \ddots & & & & \\ & & & & & \lambda_{N_\Lambda} & & & \\ & & & & & & \lambda_1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & \lambda_{N_\Lambda} \end{bmatrix}_{(N_y \cdot N_\Lambda) \times (N_y \cdot N_\Lambda)} \tag{3.16}$$

and

$$C = \begin{bmatrix} \begin{bmatrix} 1 & 1 \dots 1 \end{bmatrix} & & \\ & \begin{bmatrix} 1 & 1 \dots 1 \end{bmatrix} & \\ & & \begin{bmatrix} 1 & 1 \dots 1 \end{bmatrix}_{1 \times N_\Lambda} \end{bmatrix}_{N_y \times (N_y \cdot N_\Lambda)} \tag{3.17}$$

3.2.3 Matrix B

Having the matrices A and C, the matrix B can be optimized over the whole trajectory, allowing for multiple-step prediction, unlike the one-step prediction of the EDMD method mentioned in Section 3.1.1. The optimization problem can be formulated as

$$\min \sum_{j=1}^{M_{t,c}} \sum_{k=1}^{M_{s,c}} ||\xi\left(x_k^j\right) - \hat{y}_k\left(x_0^j\right)||_2^2, \tag{3.18}$$

where

$$\hat{y}_k\left(x_0^j\right) = CA^k z_0^j + \sum_{i=0}^{k-1} CA^{k-i-1} B u_i^j \tag{3.19}$$

for

$$z_0^j = \phi\left(x_0^j\right) \tag{3.20}$$

is output vector predicted by the matrices A,C and $\xi\left(x_k^j\right)$ is the actual output vector of k^{th} sample on j^{th} trajectory.

For optimizing over shorter interval of length M, the equation (3.19) changes as follows

$$\hat{y}_k\left(x_0^j\right) = CA_d^{k-l_k} z_{l_k}^j + \sum_{i=l_k}^{k-1} CA^{k-i-1} B_d u_i^j \tag{3.21}$$

where $l_k = \max\left(k-M, 0\right)$ and $z_{l_k} = \hat{\phi}\left(x_{l_k}^j\right)$.

The problem (3.18) can be written in matrix form

$$\min ||\Theta b - \theta||_2^2 \tag{3.22}$$

where $b = \mathrm{vec}\left(B\right)$ with $\mathrm{vec}\left(.\right)$ being a column-major vectorization of the matrix B.

$$\Theta = \left[\Theta_1^T \Theta_2^T \ldots \Theta_{M_t}^T\right]^T, \theta = \left[\theta_1^T \theta_2^T \ldots \theta_{M_t}^T\right]^T \tag{3.23}$$

for

$$\Theta_j = \begin{bmatrix} \left(u_0^j\right)^T \otimes C \\ \vdots \\ \sum_{i=l_k}^{k-1} [\left(u_i^j\right)^T \otimes \left(CA^{k-i-1}\right)] \\ \vdots \\ \sum_{i=l_k}^{M_s-1} [\left(u_i^j\right)^T \otimes \left(CA^{M_s-i-1}\right)] \end{bmatrix}, \theta_j = \begin{bmatrix} \xi\left(x_1^j\right) - CAz_0^j \\ \vdots \\ \xi\left(x_k^j\right) - CA^M z_{l_k}^j \\ \vdots \\ \xi\left(x_{M_s}^j\right) - CA^M z_{M_s-M}^j \end{bmatrix}, \tag{3.24}$$

where $\otimes$ is the Kronecker product. The problem (3.22) can be augmented as

$$\min ||\Theta b - \theta||_2^2 + \eta ||b||_2^2 \tag{3.25}$$

for some $\eta \in \mathbb{R}$. The term $||b||_2^2$ simply penalizes the magnitude of the numerical values in B, which is useful for practical applications where (3.22) might result in large numbers which are close machine precision. The matrix B can be recovered as

$$B = \mathrm{vec}^{-1}\left(b\right). \tag{3.26}$$

Chapter 4

Approximation of the Koopman operator on a singletrack model

The singletrack model described in Section 2.2.2 will be approximated by a linear Koopman system in this chapter. As described in **??**, the approximation can be done in two steps. The first step is to approximate the uncontrolled nonlinear dynamics with a matrix A and lifting functions $\boldsymbol{\phi}$. The second step uses the results from the first to find optimal B matrix.

4.1 Uncontrolled dynamics

The approximation of uncontrolled dynamics is influenced by many parameters. Most importantly, the selection of eigenvalues Λ, the number of trajectories N_T, interpolation method of $\hat{\boldsymbol{\phi}}$ and the regularization weight γ.

The number of neighbours k_D used for regularization will be considered constant $k_D = 5$ because higher values make the problem (3.8) very demanding in terms of memory and processing power.

All uncontrolled trajectories will start on an ellipse composed of states with constant kinetic energy $E_k = 500kJ$ which is an equivalent of a car weighting $1300kg$ and riding straight at $100km/h$. The ellipse is depicted in Fig. 4.1.

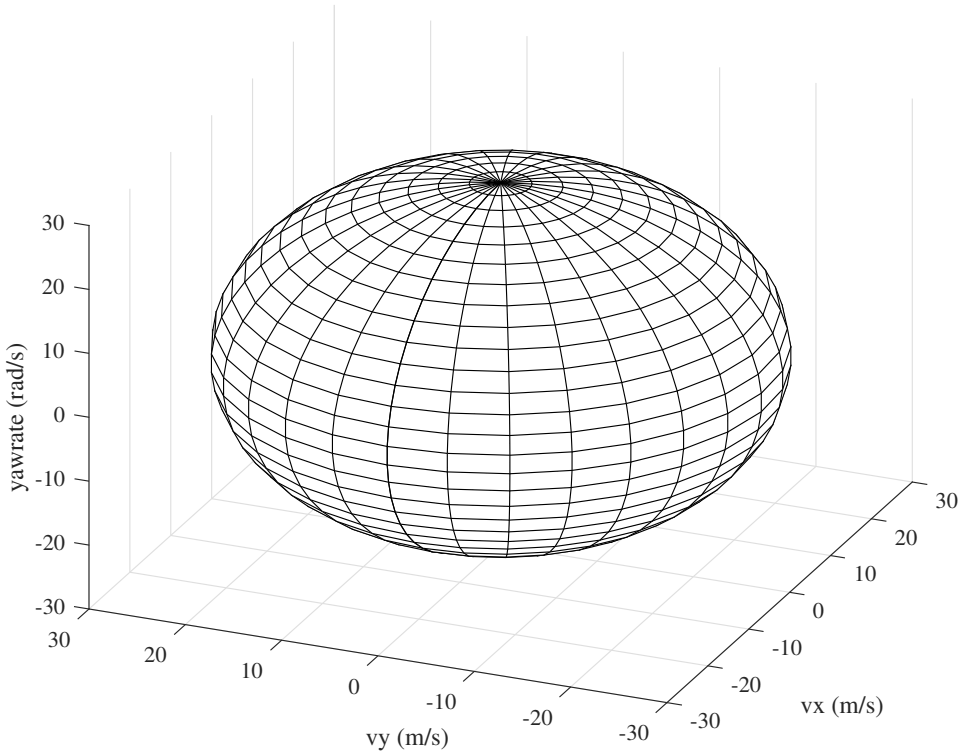


Figure 4.1: The constant-energy ellipse with $E_k = 500Kj$. All points from every Γ set of initial conditions lie on this ellipse.

The following text will evaluate the influence of previously mentioned parameters. Note that testing every possible combination would be computationally very exhausting, so a baseline set of parameters will now be established. The impact of each individual parameter will be evaluated with respect to the baseline parameter set.

The baseline parameters are shown in Table 4.1.

Parameter	Value
N_Λ	35
N_T	456
γ	0
ζ	$1e^{-8}$

Table 4.1: Baseline parameters

N_Λ is the number eigenvalues, N_T is the number trajectories and γ is the regularization cost.

The matrices A and eigenfunctions $\mathbf{\Phi}$ will be evaluated on a testing dataset T_{test} with $N_{T_{test}} = 500$. The initial conditions of T_{test} were randomly chosen inside the ellipse described above. The trajectories from T_{test} were sampled at $T_s = 0.01s$ for $T_{F,test} = 0.1s$.

The parameters will be evaluated by the values of $RMSE$ on the testing dataset T_{test} as defined in (4.1)

$$\text{RMSE} = 100\frac{\sqrt{\sum_k ||x_{koop}\left(kT_s\right) - x_{real}\left(kT_s\right)||_2^2}}{\sqrt{\sum_k ||x_{real}\left(kT_s\right)||_2^2}}, \tag{4.1}$$

where x_{koop} denotes state vector estimated by the Koopman system and x_{real} is a state vector of the nonlinear singletrack described in Section 2.2.2.

4.1.1 Selection of Λ

There is no direct approach describing the optimal selection of Λ. This work will compare three heuristic approaches and select the best one.

All three heuristics are based on Dynamic mode decomposition (DMD) described in [Sch10]. Consider dataset $X = \{x_1, x_2, ...x_K\}$ with state vectors x_k. The DMD algorithm consists in finding a matrix A_{DMD} for which it holds:

$$\begin{bmatrix} x_2 \\ x_3 \\ \vdots \\ X_K \end{bmatrix} = A_{DMD} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ X_{K-1} \end{bmatrix}. \tag{4.2}$$

The matrix can be calculated as

$$A_{DMD} = \begin{bmatrix} x_2 \\ x_3 \\ \vdots \\ X_K \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ X_{K-1} \end{bmatrix}^{\dagger}, \tag{4.3}$$

where the † symbol denotes Moore-Penrose pseudoinverse.

DMD on individual trajectories

This approach computes A_{DMD} for each trajectory individually, resulting in a set Λ_{DMD} of $N_x \cdot N_T$ eigenvalues. The set Λ is then created from the N_Λ

most dominant eigenvalues in Λ_{DMD}. The calculation of Λ_{DMD} is described in Algorithm 1.

Data: Set of trajectories T
Result: Set of eigenvalues Λ_{DMD}
for $j \leftarrow 1$ **to** N_T **do**
  $A_{DMD} = T_j\,(:, 2 : \text{end})\,/T_j\,(:, 1 : \text{end} - 1)$;
  $\Lambda_{DMD} = [\Lambda_{DMD}, \text{eig}\,(A_{DMD})]$;
end

Algorithm 1: Calculating eigenvalues for each trajectory individually.

DMD on individual trajectories without unstable eigenvalues

This approach is similar to the previous one, except that the Λ set contains only stable eigenvalues. Any unstable eigenvalues from Λ_{DMD} were shifted inside the unit circle so that their distance from the unit circle remained the same (for example $\lambda = 1.3$ would be changed to $\lambda = 0.7$).

DMD on the whole dataset

The final approach consists in applying the DMD algorithm only once but on the whole dataset instead of individual trajectories, resulting in just N_x eigenvalues in the set Λ_{DMD}. The set Λ is chosen as $\Lambda = \text{mesh}\,(\Lambda_{DMD})$ for

$$\text{mesh}\,(\Lambda) = \{\sum_{k=1}^{q} \alpha_k \lambda_k | \lambda_k \in \Lambda, \alpha_k \in \mathbb{N}, q \in \mathbb{N}, \sum_{k=1}^{q} \leq d_\lambda\}, \tag{4.4}$$

where $d_\lambda \in \mathbb{N}$ si chosen by the user. This approach was adopted from [KM18b].

4.1.2 Comparison of Λ-selection heuristics

The algorithms and their resulting Λ sets will be referred to in the order they were established according to Table 4.2.

Number	Λ set	Description
1	Λ_1	DMD on individual trajectories
2	Λ_2	DMD on individual trajectories without unstable eigenvalues
3	Λ_3	DMD on the whole dataset

Table 4.2: Ordering of compared heuristics and their Λ sets.

The eigenvalues obtained by the first approach are depicted in Fig. 4.2. The algorithm resulted in over 400 eigenvalues from which 35 most signicifant were chosen. The significance of eigenvalues was determined by a histogram as shown in Fig. 4.3.

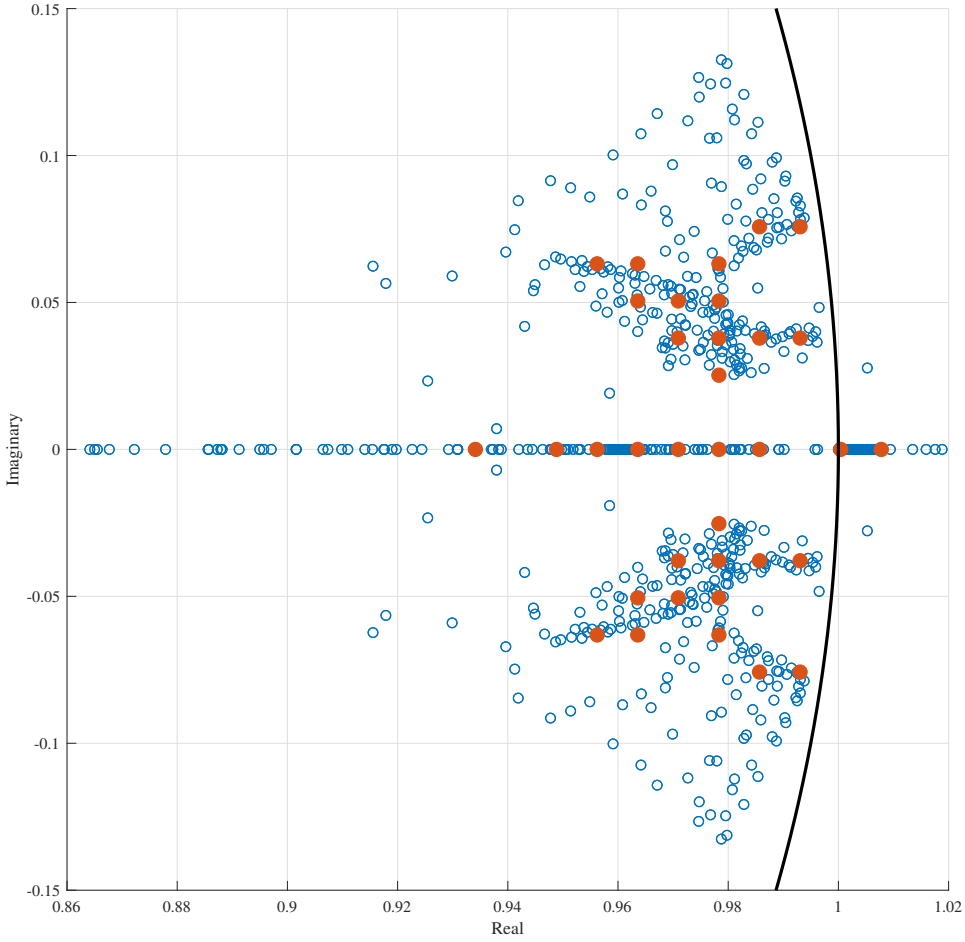


Figure 4.2: The set Λ_{DMD} is depicted by the blue circles. Red points are the eigenvalues Λ_1.

The third approach resulted in 3 real eigenvalues in Λ_{DMD} and the set of 35 eigenvalues was calculated as $\Lambda_3 = \text{mesh}\,(\Lambda_{DMD})$ for $d_\Lambda = 4$.

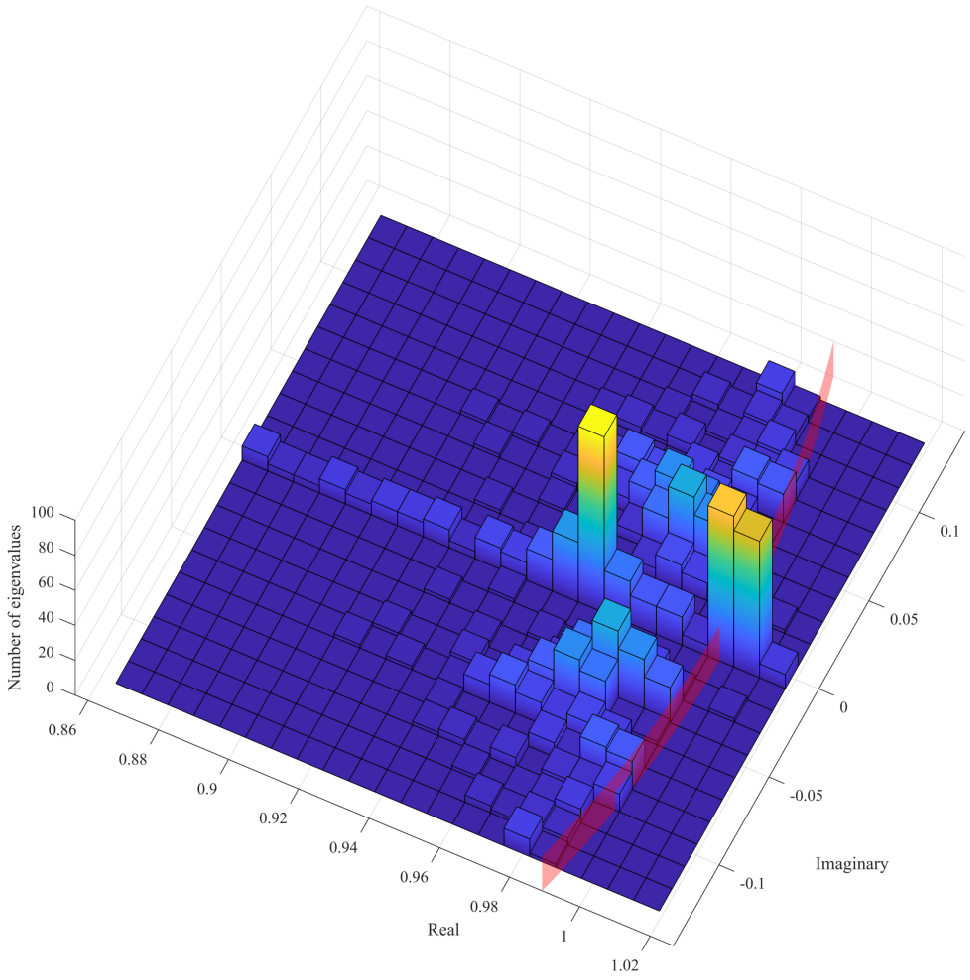


Figure 4.3: Histogram of the set Λ_{DMD} from the first and second approach. The centers of 35 squares which contained the most eigenvalues were selected as Λ_1.
The red line symbolizes the unit circle, dividing stable and unstable eigenvalues.

Comparison of the Λ sets is show in Fig. 4.4. The Λ sets were compared on the testing dataset T_{test} with $RMSE$ defined in (4.1). The results are shown in Table 4.3. The best heuristic is the approach *DMD on individual trajectories* described in Section 4.1.1.

4.1.3 Number of eigenvalues N_Λ

The influence of parameter N_Λ is investigated in this subsection. The values of N_Λ were drawn from the set $N_{\Lambda,X} = \{10, 22, 35, 51, 70, 95\}$. The values of $N_{\Lambda,X}$ were selected so that the eigenvalues were symmetrical around the real axis, as in Fig. 4.2.

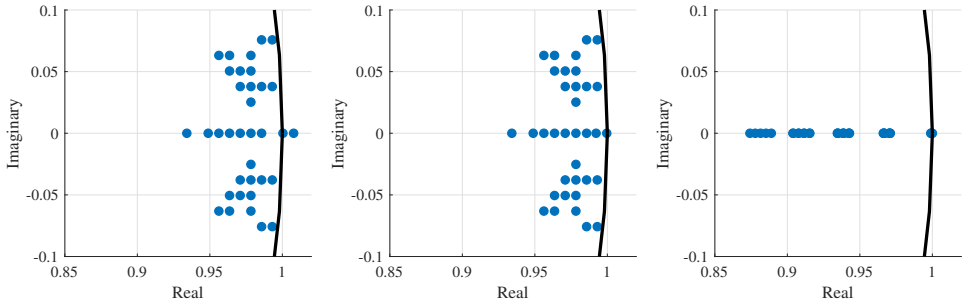


Figure 4.4: The eigenvalues from the first algorithm are depicted in the leftmost image. The middle one contains similar values with the exception of unstable eigenvalues, which were moved inside the unit circle (black line). The rightmost image contains eigenvalues Λ_3 obtained by the third approach.

	1	**5**	**10**	**15**	**20**	**30**	**50**
Λ_1	6.70	4.82	4.23	4.08	4.11	4.29	4.86
Λ_2	6.71	4.82	4.24	4.08	4.12	4.30	4.87
Λ_3	7.15	5.41	4.82	4.65	4.67	4.82	5.34

Table 4.3: Comparison of heuristics for selecting Λ for different numbers of k-neighbours used for interpolation of $\hat{\Phi}$. The best results were obtained by the first approach. The optimal number of neighbours was 15 in all 3 cases.

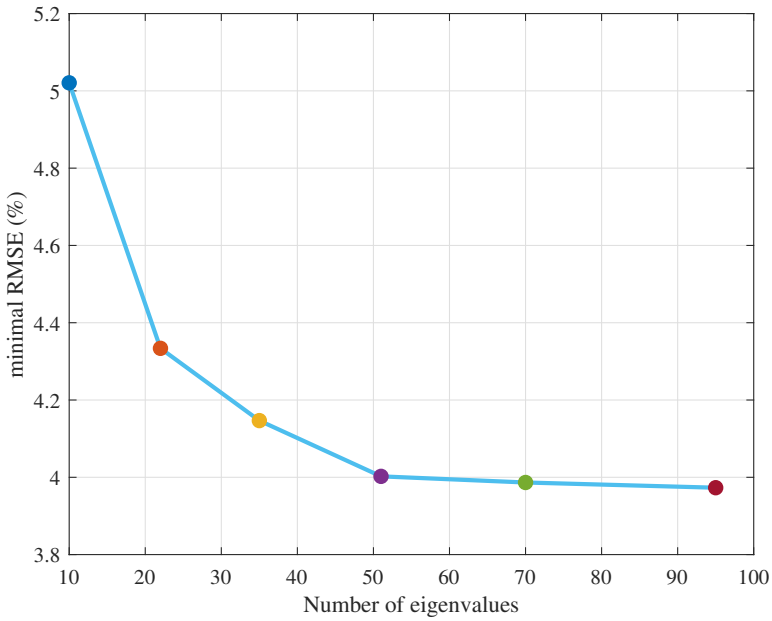
The Λ sets were selected using the heuristic approach from Section 4.1.1. The results are depicted in Fig. 4.5. The most reasonable value of N_Λ is 51, higher values do not show any significant improvement.

4.1.4 Datasize N_T

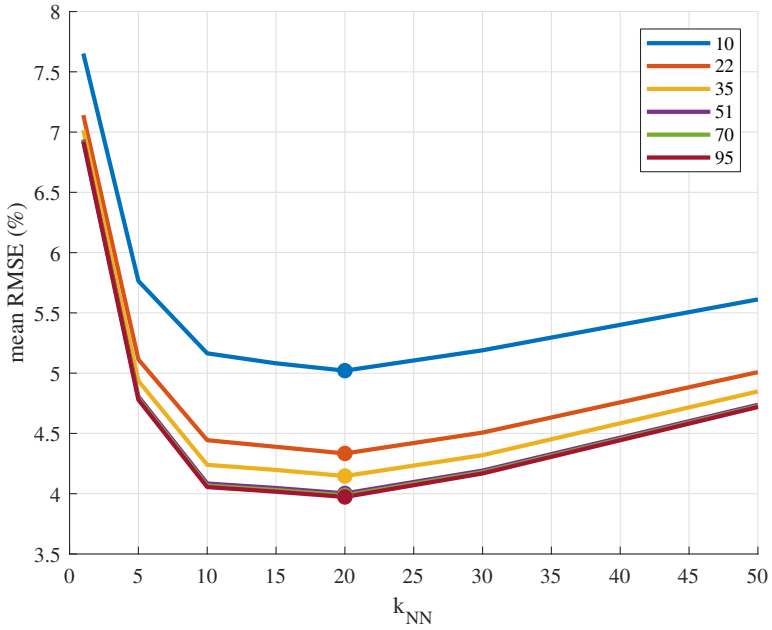
The influence of the number of trajectories N_T is investigated next. The results can be seen in Fig. 4.6. Note that the RMSE decreases steadily even for large values.

4.1.5 γ regularization

The parameter γ along with constant $k_D = 5$ did not show any improvements, as seen in Section 4.1.5. It might due to the low value of k_D. However, the size of the regularization matrix D increases with k_D, making the problem (3.8) much more demanding in terms of hardware resources.

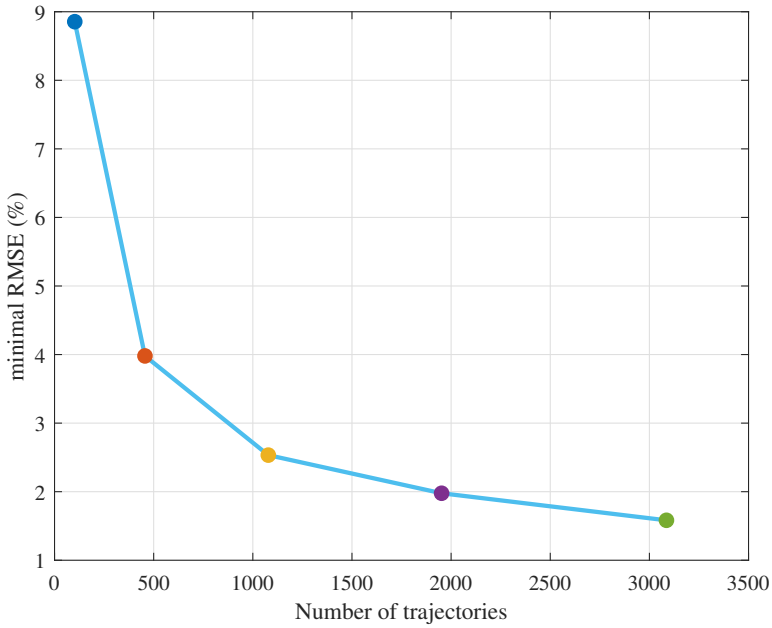


(a) : Minimal achievable mean RMSE for a given N_Λ. The colored points represent the same points as in the graph below.

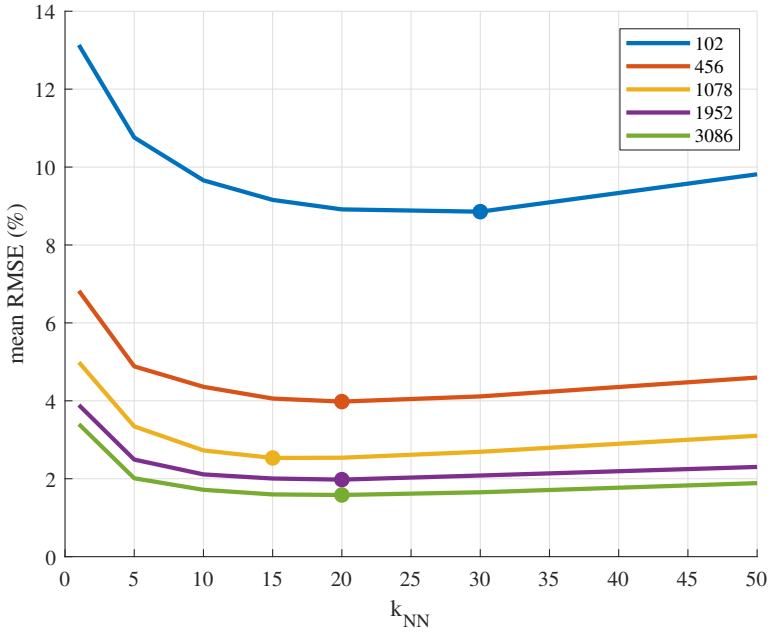


(b) : Mean RMSE for varying k-neighbours. The optimal value was 20 neighbours in all cases.

Figure 4.5: RMSE for different values of N_Λ. The colored points signify optimal values for each N_Λ, they represent the same set of points in both graphs.



(a) : Minimal achievable mean RMSE for a given N_T. The colored points represent the same points as in the graph below.



(b) : Mean RMSE for varying k-neighbours. The optimal value shifted with changing N_T.

Figure 4.6: RMSE for different number of trajectories N_T. The colored points signify optimal values for each N_T, they represent the same set of points in both graphs.

Note that this measurement was done with $\zeta = 0$ in order to prevent any influence of the other regularization term on the results.

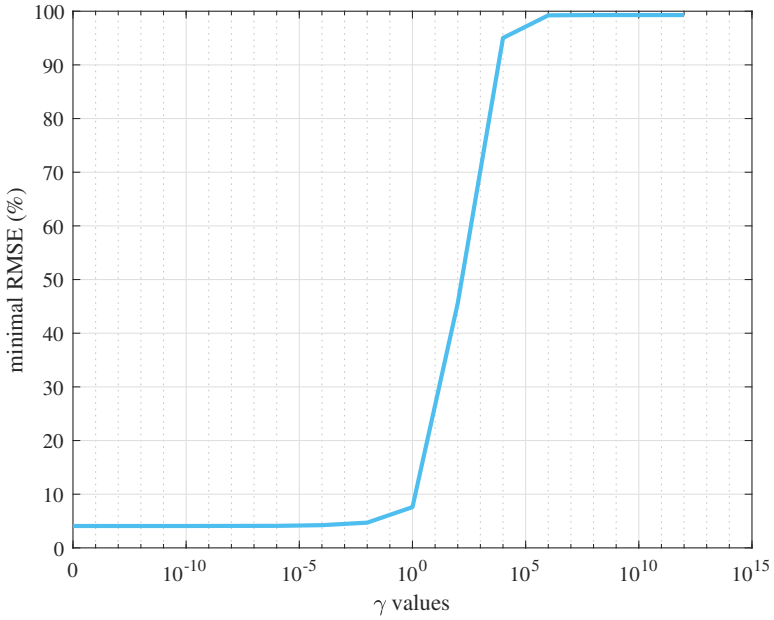


Figure 4.7: The γ parameter did not provide any improvement of the minimal achievable mean RMSE. This might be due to the fact that the nearest-neighbour interpolation implicitly performs local regularization (depending on k_{NN}) around the interpolated point, which might be sufficient for this system. Note that although the x-axis is logarithmic, the value 0 has been added to show the influence of $\gamma = 0$.

4.1.6 ζ regularization

The parameter ζ had a significant impact on the order of magnitude of vectors from the lifted space while having little to no influence on the RMSE (actually, quite similar to the γ parameter).

Usage of ζ does not have any severe impact on the dimensions of (3.8), providing a simple way to prevent possible numerical difficulties.

Note that this measurement was done with $\gamma = 0$ in order to prevent any influence of the other regularization term on the results.

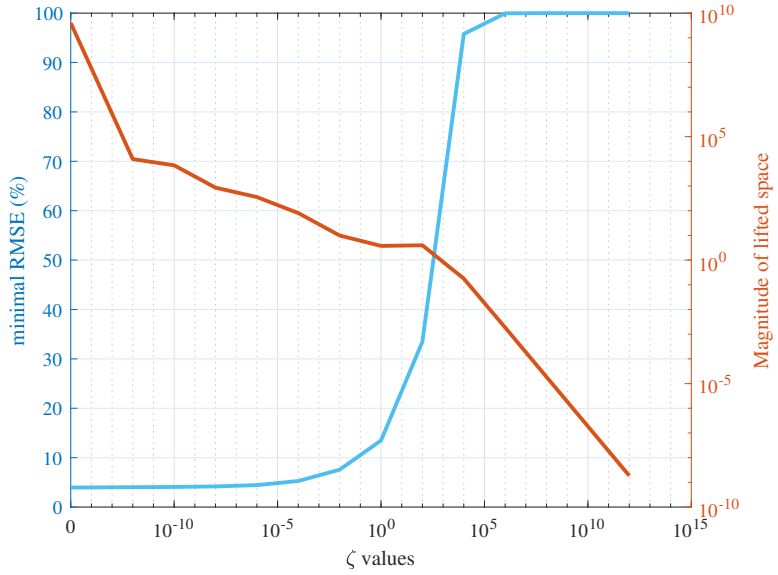


Figure 4.8: Small values of ζ have very little influence on RMSE while providing considerable decrease in the magnitudes of the vectors in the lifted space. The values 0 has been added to the logarithmic scale to show the influence of $\zeta = 0$.

4.1.7 Prediction horizon

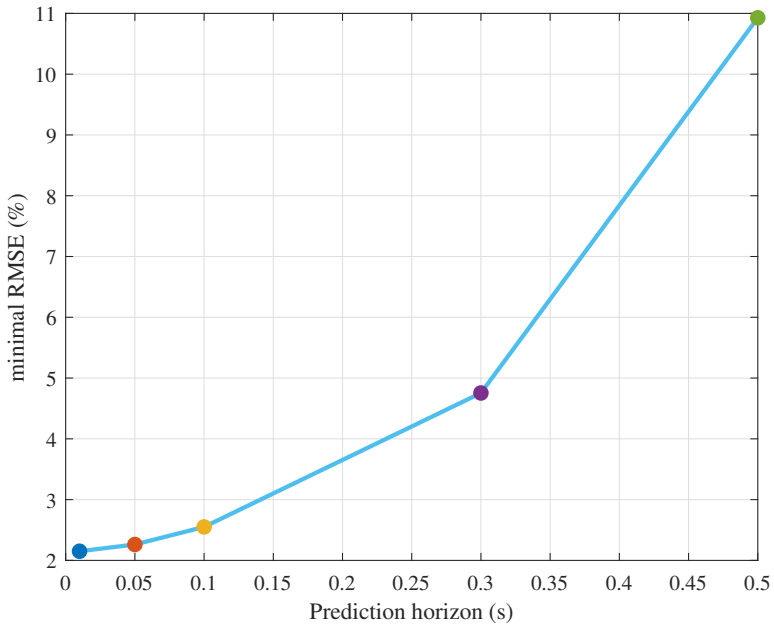
The parameters were selected based on the results of previous subsections in order to provide sufficient prediction accuracy without inflating the problem (3.8) and making it impossible or unnecessarily difficult to solve on a laptop. The final parameter set for estimation of A and $\mathbf{\Phi}$ can be found in Table 4.4.

Parameter	Value
N_Λ	51
N_T	1078
γ	0
ζ	$1e^{-12}$

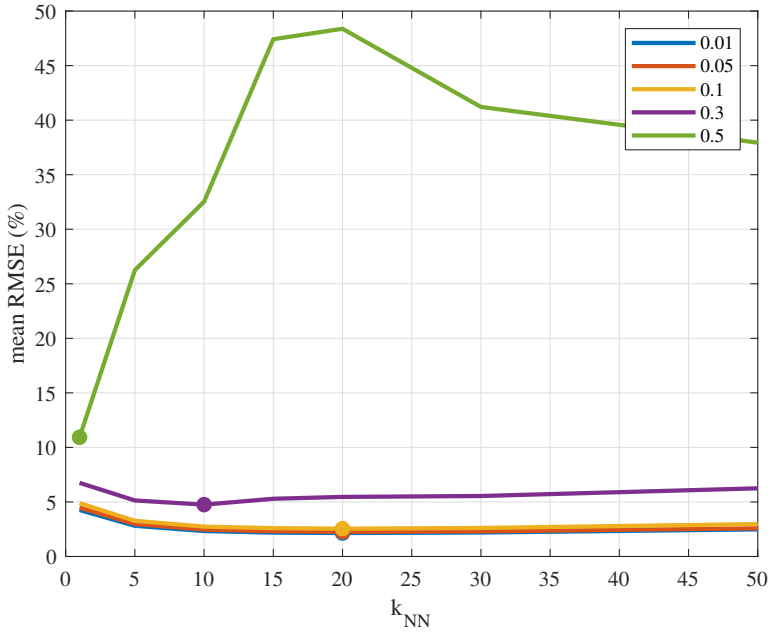
Table 4.4: Final parameters for A and $\mathbf{\Phi}$ estimation.

In order to evaluate the prediction capabilities of A and $\mathbf{\Phi}$, RMSE was calculated for varying prediction times $T_{F,test} \in \{0.01s, 0.05s, 0.1s, 0.3s, 0.5s\}$ for $T_s = 0.01s$. The results can be seen in Fig. 4.9.

The following section will use prediction horizon $T_N = 0.1s$ with mean RMSE at 2.5% and maximum RMSE 24.5%. Comparison of these values can be seen in Fig. 4.10.

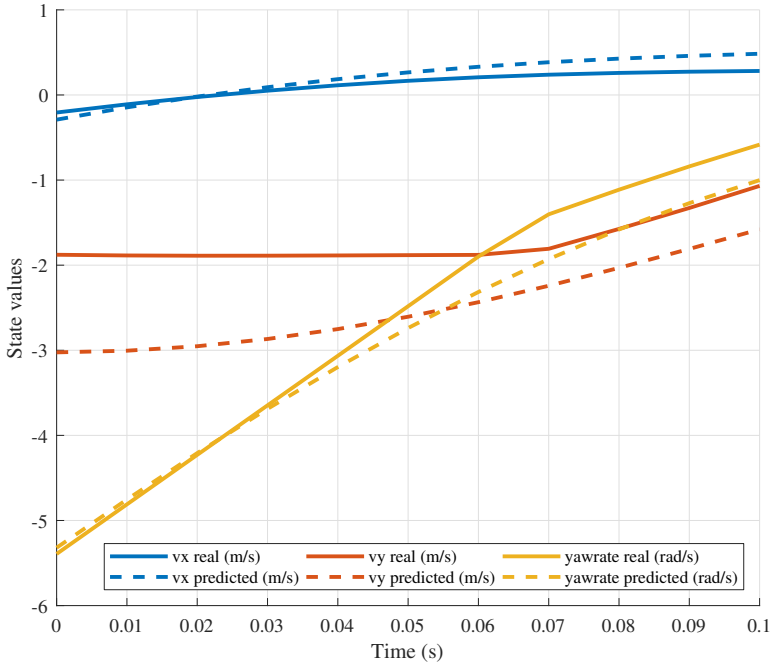


(a) : Minimal achievable mean RMSE for a given prediction horizon $T_{F,test}$. The colored points represent the same points as in the graph below.

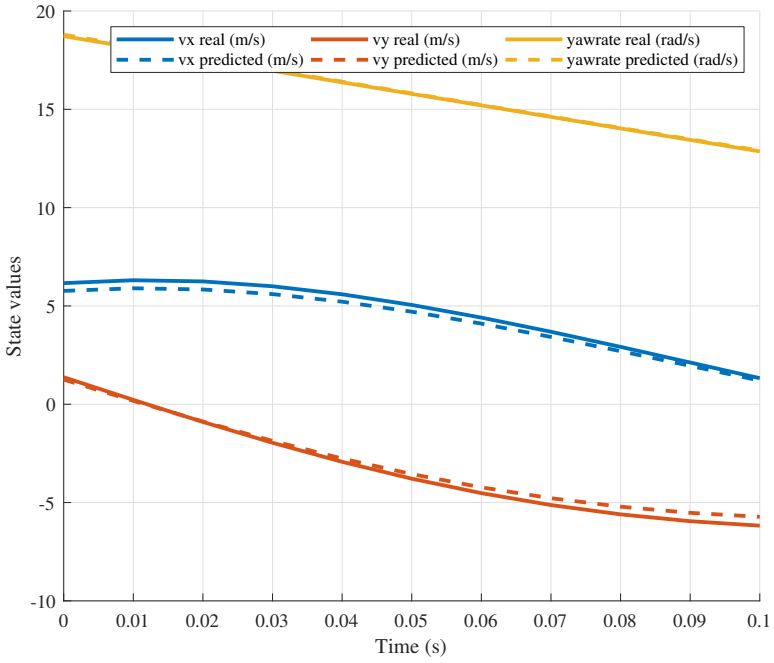


(b) : Mean RMSE for varying k-neighbours. The optimal value shifted with changing $T_{F,test}$.

Figure 4.9: RMSE for different values of $T_{F,test}$. The colored points signify optimal values for each $T_{F,test}$, they represent the same set of points in both graphs.



(a) : Trajectory with RMSE = 24.5%.



(b) : Trajectory with RMSE = 2.5%.

Figure 4.10: Comparison of RMSE = 2.5% and RMSE = 24.5% on two trajectories.

4.2 Adding control

The prediction horizon M, defined in Section 3.2.3, was chosen as

$$M = \frac{T_N}{T_s} = 10 \tag{4.5}$$

The optimization was done with 500 trajectories with random control inputs. The control inputs were uniformly distributed in following intervals:

$$\begin{array}{ll} \lambda_f & [0,0] \\ \lambda_r & [-1,1] \\ \delta_f & [-26°, 26°] \\ \delta_r & [0,0] \end{array} ,$$

where $\delta_{f/r}$ (°) denotes front/rear steering angle and $\lambda_{f/r}$ stands for front/rear longitudinal slip. The value of δ_r was chosen based on the steering angles of today's vehicles, 26° is the maximum steering angle of AUDI A4. The datasheet [aud] lists turning circle diameter 11.5m and wheelbase of 2818 mm which gives

$$\delta_f = 90° - \arctan\left(\frac{11.5/2}{2.818}\right) = 26°. \tag{4.6}$$

The values were chosen to simulate Rear-wheel drive vehicle with only front steering.

Note that the dataset size of 500 provided sufficient precision and will remain fixed. Only the influence of the regularization parameter η will be tested.

4.2.1 η regularization

The parameter η had very little influence on the prediction precision on the tested interval. The results are shown in Fig. 4.11.

The value of η used in the following chapter will be $\eta = 10^{-6}$.

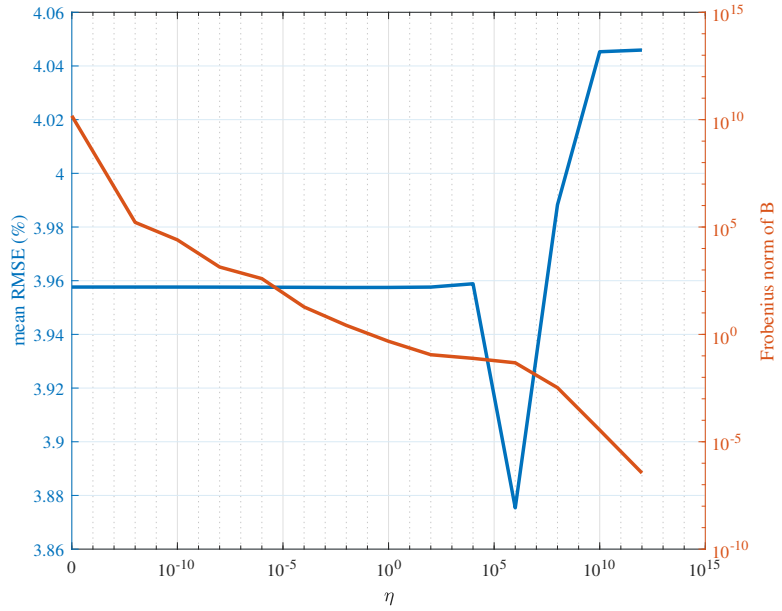


Figure 4.11: The parameter η decreased the Frobenius norm of B for almost no cost on the mean RMSE. The value 0 was added to the logarithmic scale to show the influence of $\eta = 0$.

Chapter 5

Model predictive control design

Model predictive control (MPC) originated in 1970's and quickly became very popular in chemical sectors due to its ability to handle complex dynamical system with hundreds of inputs and outputs, while respecting the system constraints . Information about history and industry applications can be found in [QB03]. Some of the strengths of MPC are the ability to include input-output constraints and to predict the plant's behaviour in the future. Most other controllers such as PID and LQR are reactive, however MPC can incorporate information about future reference and act before the reference arrives.

In automotive, MPC for nonlinear vehicle models has already been used in numerous applications. Vehicle stability control based on planar nonlinear 3DOF model in [TJ03]. Then vehicle stability control via brake torque distribution was introduced in [CG07], this works approach was based on linearizing the model around non-equilibrum operating points and it was verified on a 8DOF nonlinear vehicle model. An MPC based torque vectoring system to prevent vehicle rollover was introduced in [YZG17].

The MPC control strategy consists in solving an open-loop optimization problem over a fixed horizon of length N, resulting in N open-loop control input vectors. The first input vector is then applied to the controlled system and the whole optimization is done again at the next time-step - resulting in a closed-loop controller.

This chapter describes the linear MPC design based on the linear Koopman system established in Chapter 4. This approach was first introduced in

[KM18a] for the EDMD method referred to in Section 3.1.1 and then in [KM18b] for the method described in Section 3.1.2, which is used in this thesis.

The Koopman MPC will be compared with a MPC based on a locally linearized model.

5.1 MPC problem formulation

Linear MPC is defined as a quadratic optimization problem

$$\min_{u_k} \sum_{k=0}^{N} [(y_k - r_k)^T Qy (y_k - r_k) + u_k^T R u_k + s^T S s] \tag{5.1}$$

$$\tag{5.2}$$

$$s.t. \tag{5.3}$$

$$z_{k+1} = Az_k + Bu_k \qquad k = 0..N-1 \tag{5.4}$$

$$y_k = Cz_k \qquad k = 0..N-1 \tag{5.5}$$

$$y_{\min} - s \leq y_k \leq y_{\max} + s \qquad k = 0..N-1 \tag{5.6}$$

$$u_{\min \text{ rate}} \leq u_{k+1} - u_k \leq u_{\max \text{ rate}} \qquad k = 0..N-1 \tag{5.7}$$

$$u_{\min} \leq u_k \leq u_{\max} \qquad k = 0..N-1, \tag{5.8}$$

where Q_y,S and R are positive semidefinite cost matrices, N is the prediction horizon, $y_{\min/\max}$ are soft constraints on the output vector y_k with slack variables s and $u_{\min/\max \text{ rate}}$ are constraints on the system input rates.

Both MPC regulators were parametrized as follows:

$$Q_y = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix}, R = \begin{bmatrix} 0 & & & \\ & 100 & & \\ & & 30 & \\ & & & 0 \end{bmatrix}, S = 10^5 \cdot \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} \tag{5.9}$$

$$y_{\min} = -\begin{bmatrix} 25 \\ 2 \\ 2 \end{bmatrix}, y_{\max} = \begin{bmatrix} 25 \\ 2 \\ 2 \end{bmatrix}, u_{\min} = -\begin{bmatrix} 0 \\ 1 \\ 0.45 \\ 0 \end{bmatrix}, u_{\max} = \begin{bmatrix} 0 \\ 1 \\ 0.45 \\ 0 \end{bmatrix} \tag{5.10}$$

$$u_{\min \text{ rate}} = -\begin{bmatrix} 0 \\ 0.1 \\ 0.8 \\ 0 \end{bmatrix}, u_{\max \text{ rate}} = \begin{bmatrix} 0 \\ 0.1 \\ 0.8 \\ 0 \end{bmatrix}. \tag{5.11}$$

The scheme of the Koopman MPC is depicted in Fig. 5.1. The implementation of (5.1) was done in YALMIP [Lö].

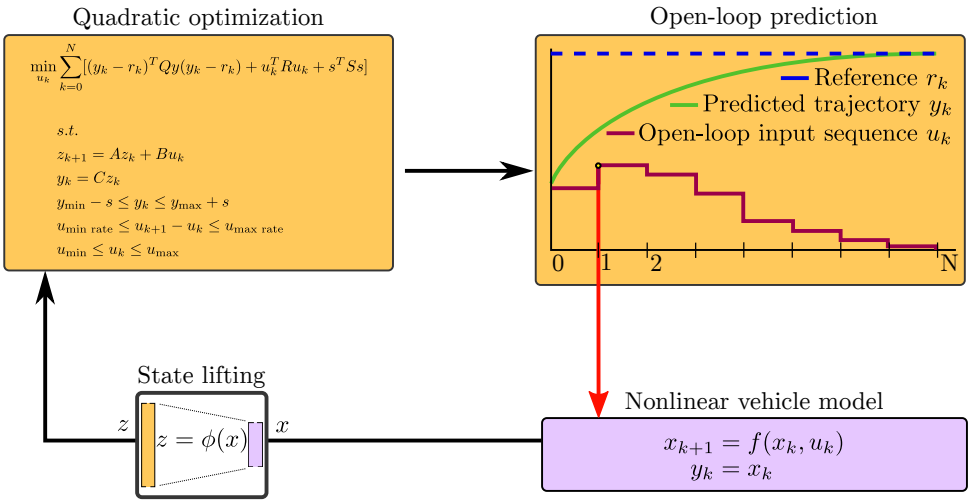


Figure 5.1: Scheme describing the Koopman MPC algorithm. Areas operating in the lifted state-space are depicted in orange color, The non-linear space is depicted in violet.

5.2 Comparison with local linearization-based MPC

This section will present comparison between Koopman MPC and MPC based on locally linearized model. The linearization was done at operating point

$$x_{\text{trim}} = \begin{bmatrix} 16.7 \\ 0 \\ 0 \end{bmatrix}, \tag{5.12}$$

meaning that the car was driving straight on with velocity $60km/h$.

The scenarios for the comparison will be constructed as follows: The initial state will always be in an unstable state where the car is well outside its linearity region. The goal will always be to stabilize the car as quickly as possible. Linearity region is meant as a subspace of the vehicle's state space where the tires exhibit linear behaviour. The studies of such regions for singletrack model were performed in [Fil18].

The reference r_k will be set as $r_k = \begin{bmatrix} -15 \\ 15 \\ 15 \end{bmatrix}$ for all tests.

Note that for simplicity, the MPC controller based on the locally-linearized singletrack model will be called *linear* controller because even though both MPC controllers are based on LTI systems, the Koopman MPC exhibits nonlinear behaviour (which is the point of the whole Koopman framework).

5.2.1 Test 1

The first test put the vehicle at arbitrary initial condition $x_0 = \begin{bmatrix} -15 \\ 15 \\ 15 \end{bmatrix}$. The results can be seen in Fig. 5.2. The Koopman MPC reached the reference much faster (approximately in $0.7s$, compared to $1.5s$) and with much less slipratio on input.

5.2.2 Test 2

In this test, the vehicle starts with initial condition at $x_0 = \begin{bmatrix} 0 \\ 15 \\ 0 \end{bmatrix}$, meaning that the vehicle is sliding sideways, to the left. The results are in Fig. 5.3. First thing to notice is that even though the car is sliding to the left, the linear controller steers to the right, unlike the Koopman MPC.

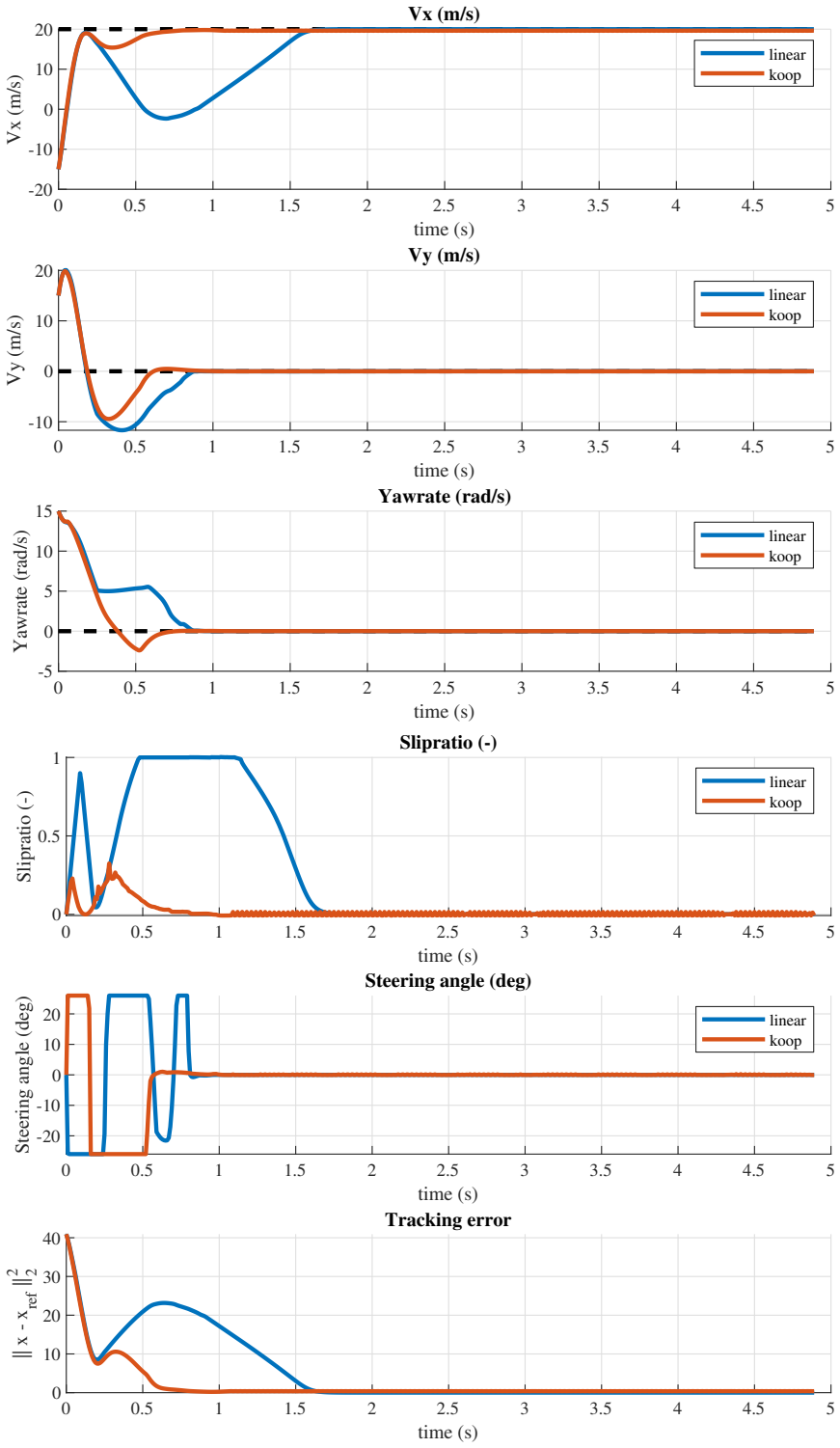


Figure 5.2: Results for $x_0 = [-15, 15, 15]^T$ on the singletrack model. In this test, the Koopman MPC was faster and more economic in terms of control action.

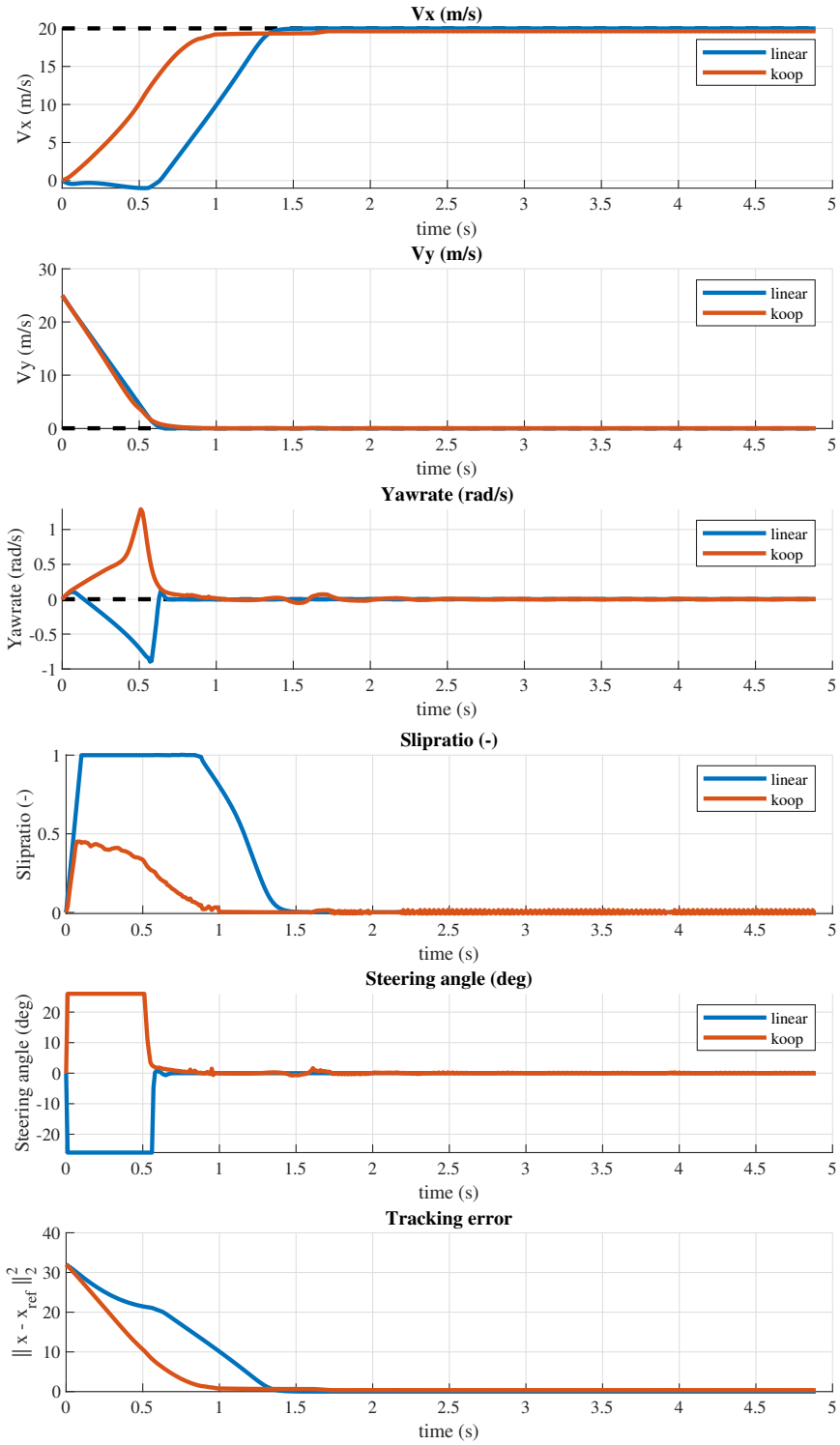


Figure 5.3: Results for $x_0 = [0, 25, 0]^T$ on the singletrack model. In this test, the Koopman MPC stabilized the system faster and with less control action.

5.2.3 Test 3

In this test, the vehicle starts with initial condition at $x_0 = \begin{bmatrix} -20 \\ 0 \\ 2 \end{bmatrix}$, meaning that the vehicle is driving backwards and turning to the right. Note that for forward-driving car positive yawrate means turning to the left, but for backwards-driving car, it means the opposite because the backwards-driving car exhibits non-minimum phase behaviour. The results are in Fig. 5.4. Not only did the Koopman MPC stabilize the system faster, it also accounted for the non-minimum phase behaviour and steered the front wheels to the right which helped to stop the turning momentum.

5.3 Validation on the twin-track model

The greatest difference between the singletrack and the full twintrack model is the phenomenon of weight transfer, which is not present in the planar singletrack. This will show the same tests done in Section 5.2 on the twintrack model.

Note that the measurements of body-fixed velocities v_x and v_y were corrected for the pitch and roll angles, because the singletrack-based controllers expect planar velocities.

5.3.1 Test 1

The first test put the vehicle at arbitrary initial condition $x_0 = \begin{bmatrix} -15 \\ 15 \\ 15 \end{bmatrix}$. The results can be seen in Fig. 5.5. Neither controller was able to stabilize the system. Both controllers exhibit oscillatory behaviour.

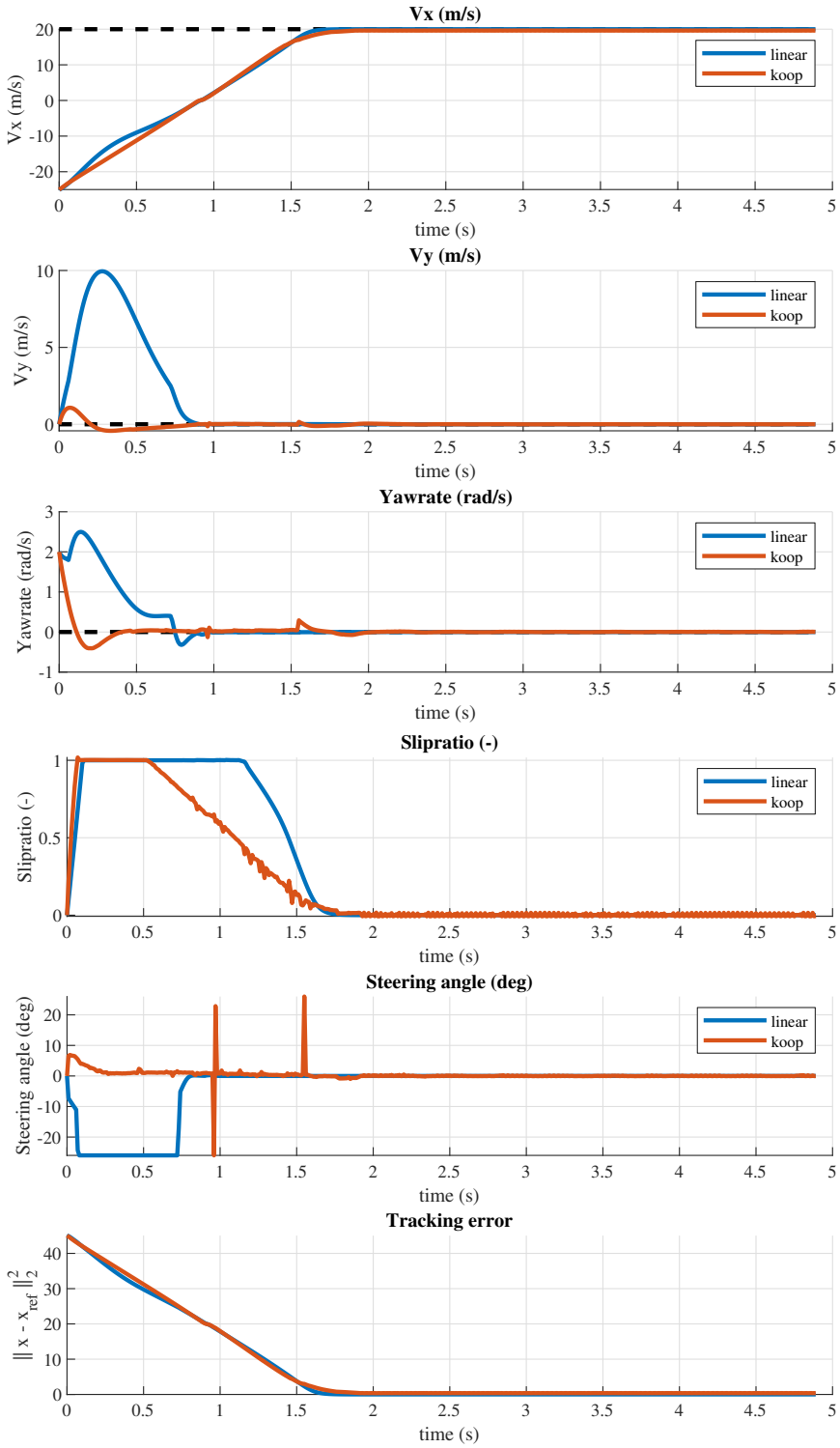


Figure 5.4: Results for $x_0 = [20, 0, 2]^T$ on the singletrack model. The Koopman MPC was faster because it accounted for the non-minimum phase behaviour of the backwards-driving vehicle.

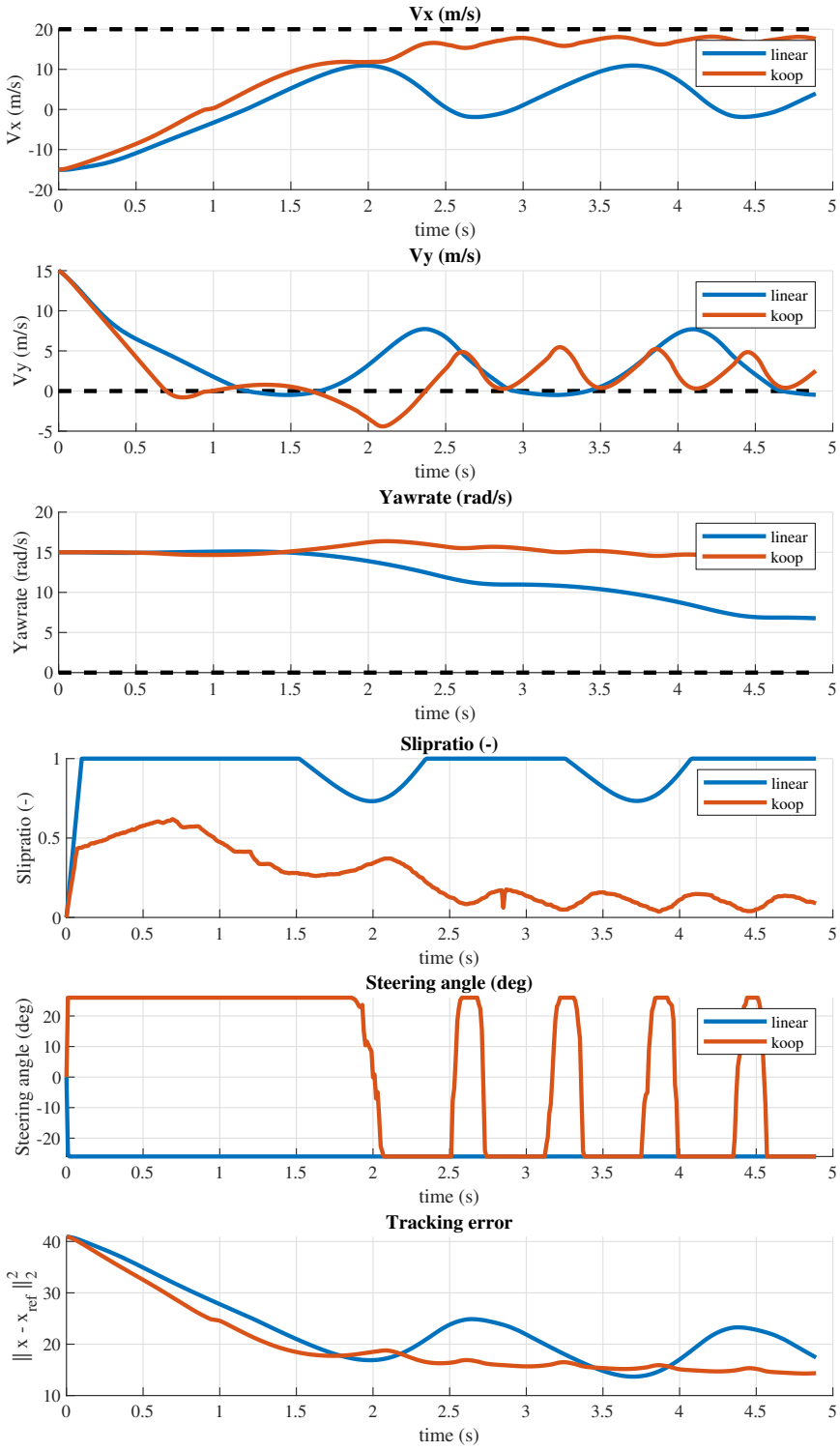


Figure 5.5: Results for $x_0 = [-15, 15, 15]^T$ on the twin-track model. Neither controller managed to stabilize the system.

5.3.2 Test 2

In this test, the vehicle starts with initial condition at $x_0 = \begin{bmatrix} 0 \\ 15 \\ 0 \end{bmatrix}$, meaning that the vehicle is sliding sideways, to the left. The results are in Fig. 5.6. The Koopman MPC managed to stabilize the system much faster. However, after stabilizing the system, the steering angle of the Koopman MPC system started oscillating, unlike the linear MPC.

5.3.3 Test 3

In this test, the vehicle starts with initial condition at $x_0 = \begin{bmatrix} -20 \\ 0 \\ 2 \end{bmatrix}$, meaning that the vehicle is driving backwards and turning to the right. Both controllers oscillate but the linear MPC manages to reach the reference state during the last second, the Koopman MPC which destabilizing the vehicle even more, considering the yawrate graph.

5.4 Summary

The Koopman MPC showed promising results on the singletrack model. The results on the validation twin-track model show that the singletrack model might not be sufficient as a control design model. Similar conclusion was presented in a paper dealing with the problems of state estimation of a racecar [HEPH16].

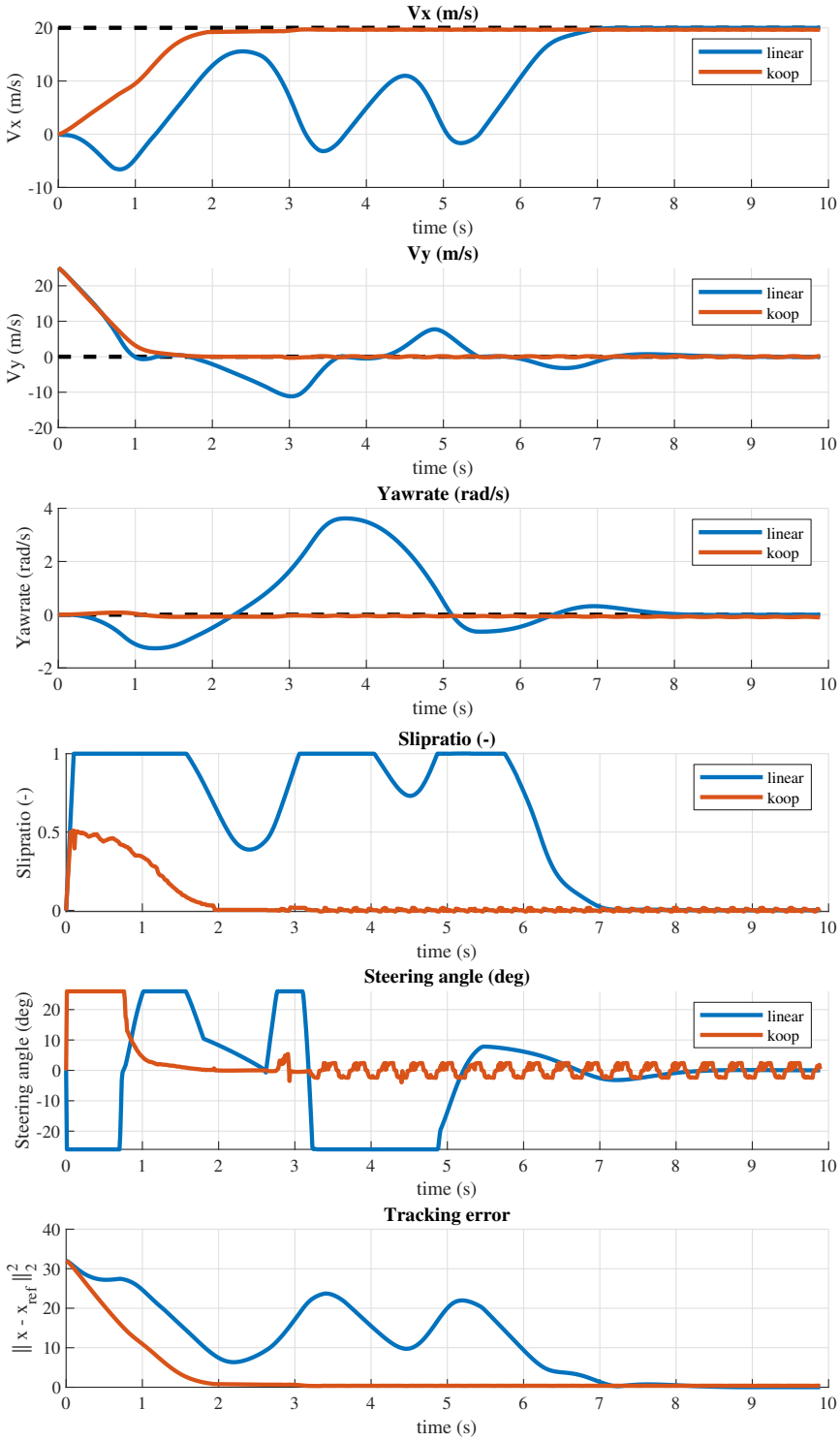


Figure 5.6: Results for $x_0 = [0, 25, 0]^T$ on the twin-track model. In this test, the Koopman MPC stabilized the system faster, but the steering angle started oscillating after the stabilization.

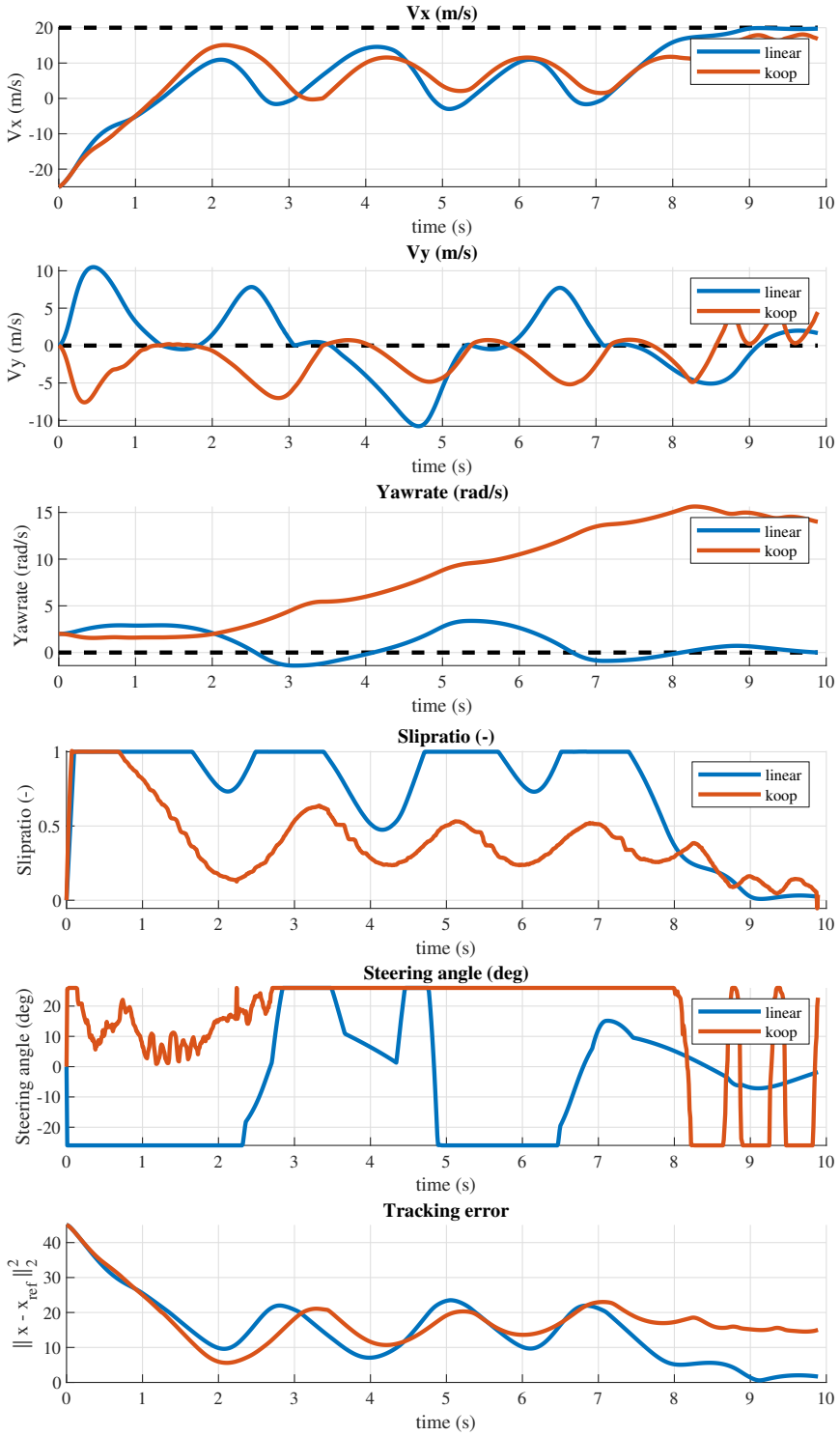


Figure 5.7: Results for $x_0 = [20, 0, 2]^T$ on the twin-track model. The Koopman MPC was unstable and did not manage to track the reference. The linear MPC managed to stabilize the system.

Chapter 6

Results

The goals of this thesis were set as follows:

1. Derive and implement nonlinear control design and validation vehicle models.
2. Create a linear representation of the control design model in a predefined operating subspace.
3. Design model predictive control algorithm based on the linear representation of a vehicle.
4. Validate the control law on a high fidelity validation model.

Their fulfillment:

1. Both vehicle models are presented in Chapter 2.
2. Linear representation is done in Chapter 4
3. The MPC design based on the Koopman predictor is presented in Chapter 5
4. The validation was performed in Section 5.3.
   The results were unsatisfactory, stimulating more research to be done in terms of validity of the singletrack model as a control design model for applications involving the nonlinear areas of vehicle dynamics.

Chapter 7

Conclusions

This thesis created a nonlinear control algorithm for control of vehicle dynamics with the usage the the Koopman operator framework, which was used to approximate the vehicle nonlinearities in a predefined subspace. The control algorithm is based on a Koopman predictor, which is an LTI system approximating the nonlinear model via multi-step prediction.
The predicted nonlinear system was a singletrack model with the full, high fidelity Pacejka tire model. The Koopman system was able to predict the singletrack trajectory for $0.1s$ with mean prediction error RMSE $= 2.5\%$ in uncontrolled scenario and with RMSE $= 4\%$ in controlled scenario.

The Koopman system was used for linear MPC design. The Koopman MPC showed good results on the singletrack model. However, it did not perform well during the validation on the full twintrack model.

Chapter 8

Future research

Future research will involve study of the differences between singletrack and twintrack model, hopefully resulting in a more suitable, low fidelity control design model.

Another topic of interest will be incorporating the input nonlinearities into the system. The current approach allows for approximating the uncontrolled nonlinearities via lifting the nonlinear states, with the control added afterwards through the linear B matrix. One way approach could be incorporating the input vector as an additional state vector, as was done in [KM18a], the question is whether this is possible while not losing the multi-step prediction capabilities, which proved to be a crucial element of the algorithm presented here in comparison with the method from [KM18a]. A comparison of these two approaches is done in [CHH].

Next, the parameter γ defined in (3.8) showed surprisingly no improvement of the prediction. The reason could be the fact that although the γ parameter performs global regularization, the interpolation method of nearest neighbours might be sufficient to locally regularize the lifted state. This could mean that the costly computations with γ regularization might not be necessary if a suitable interpolation method is used.

To sum up, my goals in the future are to improve the Koopman vehicle control framework introduced in this thesis to provide a simple yet effective framework for vehicle dynamics control. I plan on doing so during my Ph.D. studies with Ing. Milan Korda, Ph.D., the author of the Koopman model predictive approach, as my supervisor.

Appendix A

Bibliography

[aud] *TECHNICAL SPECIFICATIONS: 2018 AUDI A4 2.0 TFSI QUATTRO KOMFORT ALLROAD*, `https://www.auto123.com/en/new-cars/technical-specs/audi/a4/2018/allroad/20-tfsi-quattro-komfort/#suspension`, Accessed: 2019-05-21.

[Aut18] Rimac Automobili, *Automotive challenge*, 2018.

[CG07] Sehyun Chang and Timothy J. Gordon, *Model-based predictive control of vehicle dynamics*, International Journal of Vehicle Autonomous Systems **5** (2007), no. 1/2, 3.

[CHH] Vit Cibulka, Tomas Hanis, and Martin Hromcik, *Data-driven identification of vehicle dynamics using koopman operator*.

[Fil18] Jan Filip, *Trajectory tracking for autonomous vehicles*, 2018.

[gov68] *I. on governors*, Proceedings of the Royal Society of London **16** (1868), 270–283.

[HEPH16] Martin Haudum, Johannes Edelmann, Manfred Plöchl, and Manuel Höll, *Vehicle state estimation from a sports-car application point of view focusing on handling dynamics*, Advanced Vehicle Control AVEC'16, Crc Press, dec 2016, pp. 521–526.

[KM18a] Milan Korda and Igor Mezić, *Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control*, Automatica **93** (2018), 149–160.

[KM18b] Milan Korda and Igor Mezić, *Learning koopman eigenfunctions for prediction and control: the transient case*.

[Koo31] B. O. Koopman, *Hamiltonian systems and transformation in hilbert space*, Proceedings of the National Academy of Sciences **17** (1931), no. 5, 315–318.

[Lib16] Max Liben, *Wr-217e architecture design*, 2016.

[Lor12] Vittorio Lorenzi, *Using the pac2002tire model*, 2012.

[Lö] J. Löfberg, *Yalmip*, https://yalmip.github.io/, Accessed: 2019-05-21.

[MB04] Igor Mezić and Andrzej Banaszuk, *Comparison of systems with complex behavior*, Physica D: Nonlinear Phenomena **197** (2004), no. 1-2, 101–133.

[MM96] William F. Milliken and Douglas L. Milliken, *Race car vehicle dynamics*, Society of Automotive Engineers Inc., Great Britain, 1996.

[Pac02] Hans Pacejka, *Tyre and vehicle dynamics*, Elsevier LTD, Oxford, 2002.

[Pac12] ———, *Tire and vehicle dynamics*, Elsevier LTD, Oxford, 2012.

[QB03] Joe S. Qin and Thomas A. Badgwell, *A survey of industrial model predictive control technology*, Control Engineering Practice **11** (2003), no. 7, 733–764.

[Sch10] Peter J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, Journal of Fluid Mechanics **656** (2010), 5–28.

[SHB14] Dieter Schramm, Manfred Hiller, and Roberto Bardini, *Vehicle dynamics*, Springer Berlin Heidelberg, 2014.

[SS05] Ian Postlethwaite Sigurd Skogestad, *Multivariable feedback control*, John Wiley & Sons, 2005.

[TJ03] Petter Tondel and Tor A. Johansen, *Lateral vehicle stabilization using constrained nonlinear control*, 2003 European Control Conference (ECC), IEEE, sep 2003.

[WKR15] Matthew O. Williams, Ioannis G. Kevrekidis, and Clarence W. Rowley, *A data–driven approximation of the koopman operator: Extending dynamic mode decomposition*, Journal of Nonlinear Science **25** (2015), no. 6, 1307–1346.

[YZG17] Hongliang Yuan, Dong Zhang, and T. J. Gordon, *Road vehicle rollover prevention torque vectoring via model predictive control*, 2017 36th Chinese Control Conference (CCC), IEEE, jul 2017.

Appendix B

Parameters of used vehicle models

Vehicle body parameters

Name	Value	Unit	Description
m	1300	kg	mass of vehicle body
g	9.81	m/s^{-2}	gravitational constant
J_{xx}	200	$kg \cdot m^2$	moment of inertia in x-axis
J_{yy}	1300	$kg \cdot m^2$	moment of inertia in y-axis
J_{zz}	1400	$kg \cdot m^2$	moment of inertia in z-axis
S_z	0.25	m	vertical distance between CG and spring anchor
$wheelbase$	2.745	m	wheelbase ($l_f + l_r$)
c_w	0.18	$-$	drag coefficient
ρ	1.22	$kg \cdot m^{-3}$	air density
A	2	m	area exposed to aerodynamic forces
J_w	1	$kg \cdot m^2$	wheel moment of inertia
r	0.33	m	wheel radius
$c_{a,1,3}$	30000	N/kg	front spring stiffness
$c_{a,2,4}$	40000	N/kg	rear spring stiffness
$d_{a,1,3}$	8000	$N \cdot s/m$	front damping coefficient
$d_{a,2,4}$	8000	$N \cdot s/m$	rear damping coefficient

Tire parameters

LONGVL	16.5	LFZ0	1
LCX	1	LEX	1
LKX	1	LHX	1
LVX	1	LGAX	1
LCY	1	LMUY	3
LEY	1	LKY	1
LHY	1	LVY	1
LGAY	1	LTR	1
LRES	0	LGAZ	1
LXAL	1	LYKA	1
LVYKA	1	LS	1
LSGKP	1	LSGAL	1
LGYR	1	LMX	1
LVMX	1	LMY	1
PCX1	1.63	PDX1	1.06
PDX2	-0.0492	PDX3	-2.29
PEX1	0.5	PEX2	-0.11
PEX3	-0.06	PEX4	0
PKX1	19.7	PKX2	-0.15
PKX3	0.18	PHX1	-0.0005
PHX2	8.5e-5	PVX1	0
PVX2	0	RBX1	9.0
RBX2	-8.6	RCX1	1.131
REX1	0.081	REX2	-0.15
RHX1	-0.029	PTX1	1.98
PTX2	0.0003	PTX3	-0.31
PCY1	1.28	PDY1	-0.92
PDY2	0.22	PDY3	-4.55
PEY1	-1.1	PEY2	0.65
PEY3	-0.65	PEY4	-12.41
PKY1	-13.06	PKY2	1.77
PKY3	0.19	PHY1	0.0034
PHY2	-0.003	PHY3	0.044
PVY1	0.044	PVY2	-0.030
PVY3	-0.176	PVY4	-0.44
RBY1	6.4	RBY2	7.91
RBY3	-0.059	RCY1	1.16
REY1	0.22	REY2	0.43
RHY1	0.0007	RHY2	0.023
RVY1	0	RVY2	0
RVY3	0	RVY4	10
RVY5	1.94	RVY6	-50
PTY1	1.8	PTY2	1.8
QBZ1	8.40	QBZ2	-2.96

QBZ3	0.45	QBZ4	-0.45
QBZ5	-0.45	QBZ9	3.45
QBZ10	0	QCZ1	1.2
QDZ1	0.1	QDZ2	-0.003
QDZ3	-0.55	QDZ4	8.4
QDZ6	-0.003	QDZ7	0.005
QDZ8	-0.11	QDZ9	0.11
QEZ1	-2.9	QEZ2	-0.50
QEZ3	0	QEZ4	-0.12
QEZ5	-3.71	QHZ1	0.003
QHZ2	0.00084	QHZ3	0.158
QHZ4	0.121	QSX1	0.032
QSX2	0.49	QSX3	0.12
QSY1	0.010	QSY2	0
QSY3	0	QSY4	0
SSZ1	0.023	SSZ2	0.019
SSZ3	0.5	SSZ4	-0.28
QTZ1	0.2	MBELT	4.10