HAL open science

Learned and Hybrid Strategies for Control and Planning of Highly Automated Vehicles

Agapius Bou Ghosn

To cite this version:

Agapius Bou Ghosn. Learned and Hybrid Strategies for Control and Planning of Highly Automated Vehicles. Robotics [cs.RO]. Université Paris sciences et lettres, 2023. English. NNT : 2023UPSLM029 . tel-04250488

HAL Id: tel-04250488

https://pastel.hal.science/tel-04250488v1

Submitted on 19 Oct 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

PSL UNIVERSITÉ PARIS

THÈSE DE DOCTORAT
DE L'UNIVERSITÉ PSL

Préparée à MINES Paris

Stratégies Apprises et Hybrides pour le Contrôle et la Planification des Véhicules Autonomes
Learned and Hybrid Strategies for Control and Planning of Highly Automated Vehicles

Soutenue par

Agapius BOU GHOSN

Le 15 Septembre 2023

École doctorale n°621

Ingénierie des Systèmes, Matériaux, Mécanique, Énergétique

Spécialité

Informatique Temps Réel, Robotique et Automatique

Composition du jury :

Brigitte d'Andréa-Novel Professeure, Mines Paris	*Présidente*
David Filliat Professeur, ENSTA	*Rapporteur*
Philippe Martinet Directeur de recherche, INRIA	*Rapporteur*
Christian Gerdes Professeur, Stanford University	*Examinateur*
Philip Polack Docteur, FAIRMAT	*Examinateur*
Arnaud de La Fortelle Professeur associé, Mines Paris & CTO Heex Technologies	*Directeur de thèse*

MINES PARIS | PSL

Abstract

The advancement of autonomous vehicles represents a significant leap forward in the pursuit of safer and more reliable modes of transportation. Reaching full autonomy in vehicles has become the central focus of researchers and experts in the field in the last decade. Full autonomy demands precise representation of the vehicle's dynamics across various components within its architecture, to ensure the operation in a wide range of scenarios. To achieve this purpose, this thesis integrates the notion of learning to vehicle observers and planners.

Through the integration of hybrid and learned techniques, our objective is to greatly enhance the vehicle's capacity to accurately observe its state. Achieving precise state knowledge is critical as both the planning and control layers rely on this information. We test our observing techniques on real vehicle applications proving the ability of the proposed methods to achieve accurate observations in real-life scenarios, even at the limits of handling of the vehicle. The proposed methods present significant advantages over state-of-the-art methods.

After achieving accurate state observations, we propose a simple yet accurate hybrid model in the second stage. This model precisely describes the vehicle's behavior, allowing the development of a planner that can generate feasible trajectories, even in high-dynamic scenarios. An MPPI-based plan and control scheme is proposed and thoroughly tested across various maneuvers. Comparing our approach to the commonly used kinematic bicycle model in planning applications, our results clearly demonstrate the superiority of the proposed method. Notably, the planner utilizing our hybrid model ensures safer and more precise vehicle behavior.

This thesis demonstrates the capabilities of the proposed learned and hybrid neural network architectures in accurately representing the complex dynamics of the vehicle. Through simulated and real vehicle experiments, the proposed methods prove their ability to outperform state-of-the-art methods in observing and planning applications.

Résumé en français

Les progrès des véhicules autonomes représentent une avancée significative dans la recherche de modes de transport plus sûrs et plus fiables. Atteindre l'autonomie complète des véhicules est l'objectif principal des chercheurs dans ce domaine au cours des dernières années. L'autonomie complète exige une représentation précise de la dynamique du véhicule à travers les différents composants de son architecture, afin d'assurer le fonctionnement dans une large gamme de scénarios. Pour atteindre cet objectif, cette thèse introduit la notion d'apprentissage pour les observateurs et les planificateurs de véhicules.

Grâce à l'intégration de techniques d'hybridation entre modélisation et apprentissage, notre objectif est d'améliorer la capacité du véhicule à observer son état. L'obtention d'une connaissance précise de l'état est essentielle pour les couches de planification et de contrôle qui dépendent de cette information. Les techniques d'observation proposées sont testées sur des applications de véhicules réels, ce qui prouve la capacité des méthodes proposées à réaliser des observations précises dans des scénarios réels, même aux limites de la manipulation du véhicule. Les méthodes proposées présentent des avantages significatifs par rapport aux méthodes de l'état de l'art.

Après avoir obtenu des observations précises de l'état du véhicule, nous proposons, dans un deuxième temps, un modèle hybride simple et précis. Ce modèle décrit précisément le comportement du véhicule, ce qui permet de développer un planificateur capable de générer des trajectoires réalisables, même dans des scénarios très dynamiques. Un schéma de planification et de contrôle basés sur la technique MPPI sont proposés et testés pour diverses manœuvres. En comparant notre approche au modèle cinématique de bicyclette couramment utilisé dans les applications de planification, nos résultats démontrent la supériorité de la méthode proposée. Notamment, le planificateur utilisant notre modèle hybride garantit un comportement plus sûr et plus précis du véhicule.

Cette thèse démontre les capacités des architectures de réseaux neuronaux appris et hybrides proposées à représenter avec précision la dynamique complexe du véhicule. Grâce à des expériences sur des véhicules simulés et réels, les méthodes proposées prouvent leur capacité à surpasser les méthodes de pointe dans les applications d'observation et de planification.

Acknowledgements

I would like to express my deepest gratitude to the individuals and institutions whose unwavering support and guidance have played a crucial role in the completion of this thesis.

First of all, I would like to thank my thesis director: Arnaud, who believed in me from the very beginning. I am profoundly grateful for the knowledge, skills, and experiences you've imparted to me, which will continue to influence my academic and professional pursuits. I would like to thank as well my thesis co-supervisor: Philip, who was always there to guide me through this journey to reach great achievements. Your belief in my potential and your tireless commitment to guiding me through this academic journey have been nothing short of remarkable.

I thank my parents for their love and support since I was a little kid. I am indebted to them for all the efforts and sacrifices that led me here. Your encouragement and belief in me in the different parts of my life played a major role in achieving this thesis. I extend my thanks to my sisters who saw me in my stressful moments and were there to provide all the assistance when needed.

I would like to thank Marie Ange, with whom I've shared both the challenging and joyous moments. Your belief in me has often surpassed my own, and your optimism has been a constant source of inspiration. Your constant support, even through your own challenges, had a profound impact on my thesis.

During my thesis, I had the opportunity to test my theories on real vehicles in the Technical University of Braunschweig. I would like to thank Marcus Nolte whose great dedication in our meetings and experiments significantly contributed to the success of my mission. Furthermore, I would also like to thank Richard Schubert with whom I became friends during my time there; his support was appreciated throughout my journey.

I would like to thank Jean Pierre who I met at the lab in Mines Paris and with whom I became great friends. Jean Pierre offered me a great amount of support and guidance over the past three years. I am deeply appreciative of his friendship and the positive impact it had on my time at Mines Paris.

I would also like to extend my gratitude to my friends, who have been a constant source of motivation. Paul, my childhood friend, your support and knowledge-sharing have been a crucial part of my journey. Hussein and Najib, your moral support has kept me motivated during the most challenging times.

Many thanks to the jury members for their time and effort in evaluating my work and for providing insightful remarks.

Last but not least, I would like to thank the lab members and staff at Mines Paris who created a friendly environment: Jules, Louis, Jonas, Hugo, Sascha and everyone else. You made the long hours of research not only productive but also enjoyable.

Contents

1 Introduction **1**
1.1 Motivation . . . 2
1.1.1 Autonomous Vehicles . . . 2
1.1.2 Robotic Paradigms . . . 3
1.2 Objectives of the Thesis . . . 4
1.3 Contributions . . . 6
1.4 Context . . . 7

I Vehicle Models **9**

2 Vehicle modeling **11**
2.1 Introduction . . . 12
2.2 Four-wheel Vehicle Model . . . 13
2.2.1 Description . . . 13
2.2.2 Parameters . . . 14
2.2.3 Physics of the System . . . 16
2.2.4 Suspension Model . . . 17
2.2.5 Tire Model . . . 18
2.2.5.1 Pacejka Model . . . 20
2.2.5.1.1 Longitudinal Force (pure slip) . . . 21
2.2.5.1.2 Lateral force (pure slip) . . . 23
2.2.5.1.3 Self-aligning torque (pure side-slip) . . . 23
2.2.5.1.4 Longitudinal Force (combined slip) . . . 24
2.2.5.1.5 Lateral Force (combined slip) . . . 24
2.2.5.1.6 Self-aligning torque (combined slip) . . . 25
2.2.5.2 Dugoff Model . . . 25
2.2.5.3 Linear Model . . . 26
2.2.5.4 Discussion . . . 27
2.3 Bicycle Model . . . 28
2.3.1 Dynamic Bicycle model . . . 28

2.3.1.1 Description 28
2.3.1.2 Physics of the System 29
2.3.2 Extended Bicycle Model 30
2.3.2.1 Description 30
2.3.2.2 Physics of the System 30
2.3.3 Kinematic Bicycle Model 31
2.3.3.1 Description 31
2.3.3.2 Physics of the System 31
2.4 Point Mass Model 32
2.4.1 Description 32
2.4.2 Physics of the System 33
2.5 Conclusion 33

II State Observers **35**

3 Model-based State Observers **37**
3.1 Introduction 38
3.2 Observability 39
3.3 Types of Observers 39
3.3.1 Luenberger Observer 40
3.3.2 Extended Luenberger Observer 41
3.3.3 Sliding Mode Observer 42
3.3.4 Kalman Filter 43
3.3.5 Extended Kalman Filter 44
3.4 Application to Vehicles 45
3.4.1 Side-slip angle observers 47
3.4.1.1 Methodology 47
3.4.2 Velocity Observers 49
3.4.2.1 Methodology 49
3.4.3 Full State Observers 50
3.4.3.1 Methodology 50
3.4.4 Discussion 51
3.5 Model Validity in Observers 51
3.5.1 EKF definition 52
3.5.2 Testing data generation 53
3.5.3 EKF testing 54
3.6 Conclusion 55

4 Learning-based Observers **57**
4.1 Introduction . . . 58
4.2 Neural networks . . . 58
4.2.1 Overview . . . 58
4.2.2 Stochastic Gradient Descent . . . 59
4.2.3 Feedforward Neural Networks . . . 61
4.2.4 Convolutional Neural Networks . . . 62
4.2.5 Recurrent Neural Networks . . . 63
4.2.5.1 Definition . . . 63
4.2.5.2 Long Short-Term Memory . . . 64
4.2.6 Transformers . . . 66
4.3 Learned Observers . . . 67
4.4 CNN-based observer . . . 68
4.4.1 Considered system . . . 69
4.4.2 Data generation . . . 70
4.4.2.1 Training data generation . . . 70
4.4.2.2 Validation and testing data generation . . . 71
4.4.3 Observer architecture . . . 72
4.4.3.1 CNN-based observer . . . 73
4.4.3.2 LSTM-based observer . . . 74
4.4.4 Results and analysis . . . 74
4.4.4.1 Metric . . . 75
4.4.4.2 CNN-based observers comparison . . . 75
4.4.4.3 LSTM-based observers comparison . . . 76
4.4.4.4 Comparison with the EKF . . . 76
4.4.4.5 Discussion . . . 78
4.5 Conclusion . . . 79

5 Learned Observers Applied to Real Vehicles **81**
5.1 Introduction . . . 82
5.2 Stadtpilot vehicle description . . . 82
5.2.1 Overview . . . 82
5.2.2 Sensor setup . . . 82
5.3 Data collection . . . 85
5.4 Vehicle velocity and yaw rate observer . . . 86
5.4.1 Proposed architecture . . . 88
5.4.2 Results . . . 90
5.4.2.1 State-of-the-art observers . . . 90
5.4.2.2 Overall performance . . . 91
5.4.2.3 Low dynamic maneuver . . . 92
5.4.2.4 High dynamic maneuver . . . 93

5.4.2.5 Can an attention-based observer easily substitute the proposed solution? . . . 98
5.4.2.6 Discussion . . . 99
5.5 Vehicle side-slip angle observer . . . 100
5.5.1 Proposed architecture . . . 100
5.5.2 Results . . . 101
5.5.2.1 State-of-the-art observers . . . 102
5.5.2.2 Overall performance . . . 102
5.5.2.3 Low dynamic maneuver . . . 103
5.5.2.4 High dynamic maneuver . . . 104
5.5.2.5 Discussion . . . 107
5.6 Conclusion . . . 107

III Motion Planners **109**

6 HEBM: A Hybrid Model for Accurate Trajectory Planning **111**
6.1 Introduction . . . 112
6.2 Overview . . . 112
6.2.1 Model predictive control (MPC) . . . 113
6.2.2 Model predictive path integral (MPPI) . . . 114
6.3 The Hybrid Extended Bicycle . . . 116
6.3.1 Data generation . . . 117
6.3.2 Methodology . . . 118
6.3.2.1 Architecture definition and training . . . 119
6.4 Proposed approach . . . 120
6.4.1 MPPI architecture . . . 120
6.4.2 Low level controllers . . . 121
6.4.2.1 Longitudinal controller . . . 122
6.4.2.2 Lateral controller . . . 122
6.5 Results . . . 123
6.5.1 Metric . . . 123
6.5.2 Oval trajectory . . . 123
6.5.3 Lane change trajectory . . . 127
6.5.4 Discussion . . . 129
6.6 Conclusion . . . 129

7 Conclusion and Perspectives **131**
7.1 Conclusion . . . 132
7.2 Limitations and perspectives . . . 133
7.2.1 CNN-based observer . . . 133
7.2.2 Learned observers applied to real vehicles . . . 134

7.2.3 The hybrid extended bicycle model 134
7.2.4 Interchangeability of architectures 134

Chapter 1

Introduction

Contents

1.1 Motivation **2**

1.1.1 Autonomous Vehicles 2

1.1.2 Robotic Paradigms 3

1.2 Objectives of the Thesis **4**

1.3 Contributions **6**

1.4 Context **7**

1.1 Motivation

1.1.1 Autonomous Vehicles

An autonomous (or self-driving) vehicle is one that is able to drive itself without human intervention. The idea behind autonomous vehicles has been present for nearly a century and has been since a target for researchers. The interest in autonomous vehicles lies in the belief[1] that they will increase road safety by eliminating human error-related accidents and that they will increase mobility to elderly and disabled persons.

Advancements in the autonomous driving field accelerated in recent years: the field has been subject to large investments and a lot of research by institutions around the world trying to reach full autonomy.

The society of automotive engineers (SAE) classifies vehicle automation into 6 levels[2] detailed in Table 1.1, ranging from level 0 where the driver is controlling the vehicle and the driver support features are limited to warnings and short assistance, to level 5 where the driver does not intervene at all and the vehicle is capable of driving itself in all conditions.

Level	Description
0	The driver is in full control of the vehicle: the driver must accelerate, decelerate and steer to maintain safety. Driver support features are limited to warnings and short assistance (e.g. automatic emergency braking).
1	The driver is in full control of the vehicle. Driver support features include steering or accelerating/braking (e.g. lane centering, adaptive cruise control).
2	The driver is in full control of the vehicle. Driver support features include steering and accelerating/braking (e.g. lane centering and adaptive cruise control at the same time).
3	The driver is not driving when autonomous features are engaged but may be asked to take over. The vehicle is able to achieve autonomy under predefined conditions.
4	The driver is not driving and will not be asked to take over. The vehicle is able to achieve autonomy under predefined conditions.
5	The driver is not driving and will not be asked to take over. The vehicle is able to achieve autonomy in all conditions.

Table 1.1: Autonomy levels defined by the SAE

[1] https://www.nhtsa.gov/technology-innovation/automated-vehicles-safety
[2] https://www.sae.org/standards/content/j3016_202104/

After years of research in autonomous driving, reaching level 5 of automation is not yet achieved. Problems arise on several levels of the autonomous vehicle. These problems include the accurate knowledge of the state of the vehicle, and the planning and control during challenging maneuvers. To build an understanding of these limitations, the different technical levels governing the behavior of the vehicle will be explored.

1.1.2 Robotic Paradigms

As any other robotic system, autonomous vehicles operate on the basis of three primitives: sensing, planning and acting. These primitives are the core of several functioning schemes defining the relationships between a robot's components. These schemes are referred to as paradigms (i.e. a set of patterns).

The primitives governing the behavior of a robotic system are defined as follows:

- **Sense primitive**: It is responsible for collecting data from available sensors. Sensors are able to perform measurements related to the internal state of the vehicle (e.g. inertial measurement unit, wheel encoder) or to the surrounding environment (e.g. camera, LIDAR, RADAR). The collected data is analyzed and used to build an understanding of the inner vehicle state and the surrounding environment.

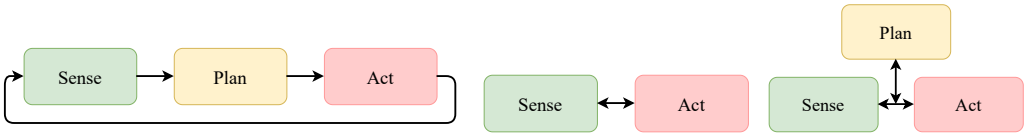
- **Plan primitive**: It is responsible for computing optimal reference trajectories that allow the vehicle to progress towards its goal state from its current state.

- **Act primitive**: It is responsible for executing the planned trajectories using the vehicle's actuators.

The presented primitives can be employed in three robotic paradigms [Murphy 2000]: a hierarchical paradigm, a reactive paradigm and a hybrid paradigm. These paradigms define the interactions between the different primitives.

The hierarchical paradigm, shown in Figure 1.1a consists of sensing, then planning, then acting. Thus, the process takes a sequential form starting by capturing sensory information, feeding them to the planner where the reference trajectory is calculated, and actuating according to the planned instructions. Although this architecture describes the order and the relationships between the different primitives, it introduces a bottleneck between sensing the real world and executing the actions which could result in inaccuracies.

The reactive paradigm shown in Figure 1.1b couples the Sense and Act primitives, the sensory information is presented directly to the actuators which results in faster action coping with the real world. Although this architecture solves the



(a) Hierarchical Paradigm (b) Reactive Paradigm (c) Hybrid Paradigm

Figure 1.1: Robotic Paradigms

disadvantages of the hierarchical architecture, the absence of the planning primitive introduces new issues: the robot is not able to have long-term goals, neither it is able to compute optimal trajectories.

The hybrid paradigm shown in Figure 1.1c solves disadvantages introduced by previous architectures. It allows the Plan primitive to act independently from the Sense-Act coupling letting the planner compute the next goal while the robot is trying to achieve the current goal; thus, combining the advantages of both the hierarchical and reactive architectures.

All of the mentioned paradigms should be able to operate in all conditions to reach full vehicle autonomy: an autonomous vehicle should sense, plan and act at all times.

1.2 Objectives of the Thesis

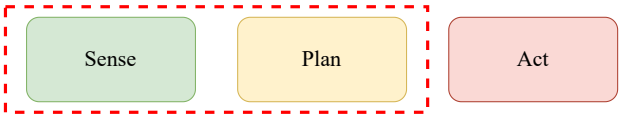


Figure 1.2: Focus of this thesis. The research targets state estimation in the Sense primitive and vehicle modeling in the Plan primitive.

As mentioned in the previous section, all of the robotic primitives are important for the functioning of an autonomous vehicle. As shown in Figure 1.2, this thesis focuses on the state estimation part of the Sense primitive; then briefly visits the Plan primitive. The thesis will emphasize on the operation of the mentioned primitives in high dynamic maneuvers where nonlinearities dominate the behavior of the vehicle's dynamics, providing methods to overcome the limitations of deterministic vehicle models. High dynamic maneuvers, associated with high accelerations, are the result of unexpected events that occur when driving and handling these events can be key to avoiding crashes.

The knowledge of the states of the vehicle at the Sense phase is crucial for the functioning of the Plan and Act primitives. Planning and acting based on inaccurate state knowledge results in inaccurate vehicle behavior. The knowledge

of non-measurable states of a vehicle and the filtering of noisy measured ones requires the implementation of State Observers. The design choice of state observers falls into one of two types: model-based observers and learning-based observers. On the one hand, model-based state observers rely on a physical model describing the vehicle dynamics and make use of the available inputs and measurements to output the state estimations. The quality of the state observations is highly dependent on the model describing the system that is expected to accurately describe the behavior of the vehicle. On the other hand, learning-based approaches are model-independent, they depend on data collected from the vehicle to train a set of neural networks used to estimate the needed quantities.

When observing in high dynamic maneuvers, model-based state observers face challenges related to the unreliability of simplified vehicle models, while complex models require the knowledge of more parameters and are associated with higher computation complexity. Learning-based state observers present better performance that depends on the quality of the training data and the neural network architecture used.

This thesis explores different deep learning architectures aiming to deliver accurate vehicle state estimations while carefully crafting the used datasets. In the first stage, proposed learned observers and data generation algorithms are evaluated on simulated vehicle data, compared to model-based observers. In the second stage, proposed learned observers are evaluated on real-world vehicles and compared to both model-based and learning-based observers for low and high-dynamic maneuvers.

After creating robust state observers, the Plan primitive is visited. The motion planning layer is responsible for generating reference trajectories for the vehicle to follow. It searches for the optimal reference trajectory given a set of criteria and safety constraints using a model of the vehicle. One important property of the planning layer is its consistency with the Act (or control) layer. In other words, the planned reference trajectories should be feasible by the vehicle, thus the used model to plan the trajectories should be able to describe the vehicle's dynamics in all scenarios.

This thesis explores the integration of learning methods into a simple vehicle model for accurate behavior description in high dynamics.

In brief, the objective of this thesis is to use learned and hybrid approaches to capture the dynamics of the vehicle. The mentioned objective is split into two main areas: first, the observing part of the vehicle is addressed while focusing on delivering accurate estimations. Second, the planning part of the vehicle is addressed while focusing on accurate modeling of the vehicle's state evolution. Both objectives consider the vehicle's operation in low dynamics and high dynamics.

1.3 Contributions

Given the objectives presented in the previous section, we present the contributions of this thesis.

As this work addresses the issues associated with deterministic vehicle models, a review of different vehicle models along with the simplifications used in the literature will be presented in Part I. Afterward, the work addresses state observers in Part II. A review of model-based observers that depend on the presented vehicle models is introduced in chapter 3. Learning-based observers are then explored; different learning-based observer architectures are proposed and are detailed as follows:

- A learning-based observer is introduced in chapter 4 aiming to filter GPS measurements to estimate the position and heading of a vehicle. The developed approach is applied to the kinematic bicycle model and is compared to an Extended Kalman Filter[3]. In this work, the data generation algorithms are detailed and the tests are carried out for different levels of noise using a simulator. This work shows the limited capabilities of model-based observers and promotes the interest in learning-based methods.

- A learning-based observer is introduced in chapter 5 aiming to estimate the vehicle's longitudinal and lateral velocities and yaw rate. The developed approach is applied to real vehicles. It makes use of the vehicle's conventional sensors to deliver the estimations while comparing the results to accurate ground truth sensors. In this work, the data collection is carefully performed to include different types of maneuvers. The developed observer is then tested in city-driving and dynamic maneuvers to prove its robustness. A comparison with state-of-the-art methods gives an insight into the performance of the developed approach.

- A hybrid observer involving vehicle's kinematics and neural networks is also introduced in chapter 5 to estimate the vehicle's side-slip angle[4]. This approach is applied to real vehicles as well. It takes advantage of a model-based side-slip calculation and uses a learned architecture to account for its errors. The approach is trained and tested on the same vehicle datasets mentioned earlier. A comparison with other model-based and learning-based methods provides insight into the performance of the method.

After dealing with observers, the thesis targets vehicle modeling in high dynamics in Part III. A hybrid vehicle model is developed in chapter 6. It is an

[3]An Extended Kalman Filter is a variant of the optimal model-based Kalman Observer detailed in chapter 3.

[4]The side-slip angle is the angle between the vehicle's velocity vector and its longitudinal axis at the center of gravity. It is highlighted in chapter 2.

augmentation of the extended bicycle model presented in the literature. The extended bicycle model considers wheel slips in a kinematic bicycle model[5]. The proposed approach makes use of neural networks to estimate the wheel slip angles given the previous vehicle state. This concept makes it possible to plan accurate trajectories even when in high dynamics.

1.4 Context

This thesis was funded by the Chair Drive For All which aims to improve the international research focusing on autonomous vehicles. It was conducted in the Centre for Robotics (CAOR) at Mines Paris, PSL University. The work was done under the direction of Prof. Arnaud de La Fortelle and was co-supervised by Dr. Philip Polack.

[5] A kinematic bicycle model is a bicycle simplification of the vehicle model. It is detailed in chapter 2.

Part I

Vehicle Models

Chapter 2

Vehicle modeling

Contents

2.1 Introduction **12**

2.2 Four-wheel Vehicle Model **13**

2.2.1 Description 13

2.2.2 Parameters 14

2.2.3 Physics of the System 16

2.2.4 Suspension Model 17

2.2.5 Tire Model 18

2.3 Bicycle Model **28**

2.3.1 Dynamic Bicycle model 28

2.3.2 Extended Bicycle Model 30

2.3.3 Kinematic Bicycle Model 31

2.4 Point Mass Model **32**

2.4.1 Description 32

2.4.2 Physics of the System 33

2.5 Conclusion **33**

2.1 Introduction

As mentioned in the previous chapter, this thesis targets the Sense and Plan stages of the vehicle's functioning. The objectives of the thesis are to achieve accurate observations of the state of the vehicle at the Sense level; and to predict accurately the behavior of the vehicle at the Plan level. Vehicle models are at the basis of model-based observers and model predictive planners. For this purpose, vehicle models should be discussed.

The implementation of complex vehicle models delivers an accurate description of the behavior of the vehicle especially in high-dynamic maneuvers but it necessitates the knowledge of many vehicle parameters and results in computational deficiency; while the implementation of simplified vehicle models is computationally efficient, and necessitates less parameter knowledge but puts the model's accuracy in jeopardy. Finding an ideal model able to represent the behavior of the vehicle remains an unresolved issue that we will discuss below.

In this chapter, we will follow a sequential vehicle model presentation, starting from complex models to simplified models while highlighting the simplifications that lead to each model. The chapter will start by presenting a high-accuracy, four-wheel vehicle model in Section 2.2. The introduction of the four-wheel vehicle model will include the carbody dynamics and the tire dynamics. The tire dynamics section will include different tire models with different complexities; these can be used for the four-wheel model and its simplifications. Simplifications of the four-wheel model will lead to the dynamic bicycle model in Section 2.3.1, the extended bicycle model in Section 2.3.2, the kinematic bicycle model in Section 2.3.3, and the point mass model in Section 2.4.

In the following description, the vehicle will be simplified to a solid body, except for the wheels. The following hypotheses govern the model:

Hypothesis 1. The considered system is formed by a vehicle, which is assumed to be a rigid body having 6 degrees of freedom in addition to the four wheels.

Hypothesis 2. All the external forces applied to the vehicle are generated either from the wheels/road interaction or the aerodynamic forces. The aerodynamic forces are represented by one force applied in the opposite direction of the car's longitudinal velocity.

Hypotheses of each model are stated in the corresponding sections.

The explanations introduced in this chapter are based on the descriptions presented in [Rajamani 2012] and [Polack 2018].

Remark 1. The following dynamics equations will be expressed in the vehicle's frame (x, y, z) at the vehicle's center of gravity as shown in Figure 2.1 in contrast to the ground inertial frame (X, Y, Z); unless stated otherwise.

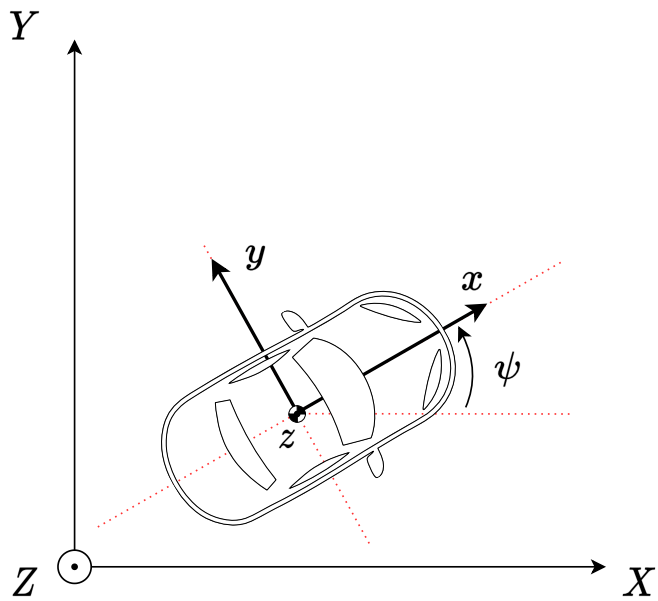


Figure 2.1: Top view showing the vehicle's frame (x, y, z) and the fixed ground inertial frame (X, Y, Z).

2.2 Four-wheel Vehicle Model

As stated in the previous section, the model presentation will follow a decreasing complexity order; thus, we start this chapter by introducing a four-wheel vehicle model. The four-wheel vehicle model will be described in Section 2.2.1, its parameters will be presented in Section 2.2.2 and its equations will be derived in Section 2.2.3. The model dynamics will impose the presentation of the suspension model in Section 2.2.4 and the tire models in Section 2.2.5.

2.2.1 Description

The four-wheel vehicle model is described in this section to be analyzed in the following sections. The four-wheel model is a representation of a vehicle's chassis. The model shown in Figure 2.2 has four wheels, two front steerable wheels, and two rear non-steerable wheels. The car's body is described by its three-dimensional position and its orientation; each wheel is described by its position from the center of gravity and its steering angle. The motion of the model is represented by the velocity vector issued from the center of gravity of the vehicle. The notations employed when representing the vehicle model are presented in Table 2.1.

The parameters describing the vehicle are presented next.

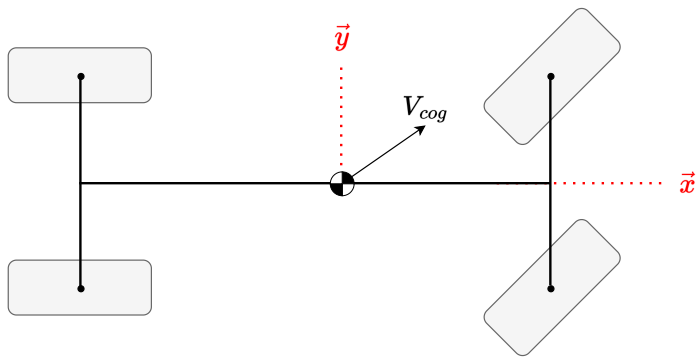


Figure 2.2: Vehicle Chassis

Notation	Description
cog	Center of gravity
$aero$	Aerodynamic
fl	Front left
fr	Front right
rl	Rear left
rr	Rear right

Table 2.1: Notations of the four-wheel model.

2.2.2 Parameters

We present the parameters of the vehicle as a step towards its motion analysis. The parameter presentation is split into constant vehicle parameters, state variables and forces. The different parameters are presented in Table 2.2.

The slope angle of the road, the height of the center of gravity, the height of the aerodynamic force, and the length between the wheels and the center of gravity are shown in Figure 2.3.

Remark 2. The mass of the vehicle, the inertia of the vehicle, as well as the position of the center of gravity, can change with the modification of the contents of the vehicle (e.g. passengers, luggage); but they are assumed to be constants when the motion of the vehicle is initiated.

Having described the model and presented its parameters, its physics will be studied next.

Constants	**Characteristics**	**Unit**
M_T	Total mass of the vehicle	kg
M_S	Suspended mass of the vehicle	kg
I_x, I_y, I_z	Inertia of the vehicle around its roll, pitch and yaw axis	$\mathrm{kg\,m^2}$
l_f	Length from the center of gravity of the vehicle to the front wheels	m
l_r	Length from the center of gravity of the vehicle to the rear wheels	m
t_l	Length from the center of gravity of the vehicle to the left wheels	m
t_r	Length from the center of gravity of the vehicle to the right wheels	m
h_{cog}	Height of the center of gravity	m
h_{aero}	Height at which the aerodynamic drag force is applied	m
Variables	**Characteristics**	**Unit**
Θ_s, Θ_b	Slope and bank angles of the road	rad
T_i	Torque applied at wheel i	N m
δ	Angle of the front wheel with respect to the axis of the vehicle (steering angle)	rad
β	Angle between the velocity vector at the center of gravity and the vehicle's longitudinal axis (side-slip angle)	rad
α_i	Slip angle at wheel i	rad
θ, ϕ, ψ	Roll, pitch and yaw angles	rad
V_x, V_y, V_z	Longitudinal, lateral and vertical velocities of the center of gravity in the vehicle frame	$\mathrm{m\,s^{-1}}$
Forces	**Characteristics**	**Unit**
F_x^i, F_y^i	Longitudinal and lateral forces applied on wheel i expressed in the vehicle's frame	N
F_z^i	Vertical force at wheel i	N
F_s^i	Suspension force at wheel i	N
F_{aero}	Aerodynamic force applied to the vehicle	N

Table 2.2: Parameters of the four-wheel model.

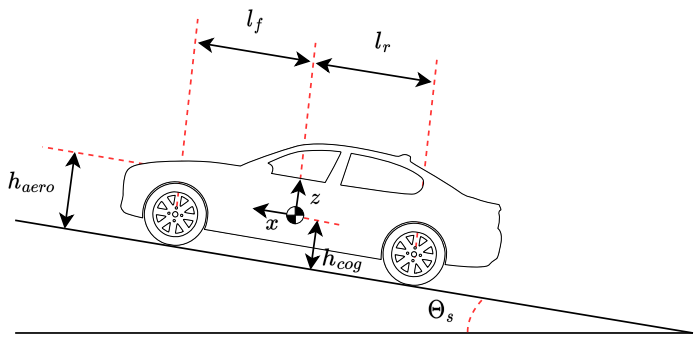


Figure 2.3: Side view of the vehicle on a road with a slope.

2.2.3 Physics of the System

The equations describing the motion of the four-wheel vehicle model are derived in this section based on the hypotheses presented earlier. The model is assumed to be on a road with a slope angle Θ_s and a bank angle Θ_b.

The second law of Newton is applied to derive the equations governing the motion of the model. As stated earlier, all the equations are derived in the vehicle's frame. The aerodynamic forces are represented by a single force applied at a height h_{aero} with a magnitude $F_{aero} = \frac{1}{2}\rho C_d A_F (V_x + V_{\text{wind}})^2$ with ρ being the mass density of the air, C_d the aerodynamic drag coefficient, A_F the frontal area of the vehicle, V_x the longitudinal vehicle's velocity and V_{wind} the wind's velocity. The detailed model with the forces applied to it is shown in Figure 2.4.

The control inputs to the model are assumed to be the wheel torques T and the steering angle δ.

The following equations describe the evolution of the longitudinal, lateral and vertical velocities:

$$\begin{aligned} M_T\dot{V}_x &= M_T\dot{\psi}V_y + (F_x^{fl} + F_x^{fr} + F_x^{rl} + F_x^{rr}) + \\ &\quad M_T g \sin(\phi - \Theta_s) - F_{aero}\cos\phi && (2.1a) \\ M_T\dot{V}_y &= -M_T\dot{\psi}V_x + (F_y^{fl} + F_y^{fr} + F_y^{rl} + F_y^{rr}) - \\ &\quad M_T g \sin(\theta - \Theta_b)\cos(\phi - \Theta_s) && (2.1b) \\ M_S\dot{V}_z &= M_S\dot{\phi}V_x + (F_z^{fl} + F_z^{fr} + F_z^{rl} + F_z^{rr}) - \\ &\quad M_S g \cos(\theta - \Theta_b)\cos(\phi - \Theta_s) + F_{aero}\sin\phi && (2.1c) \end{aligned}$$

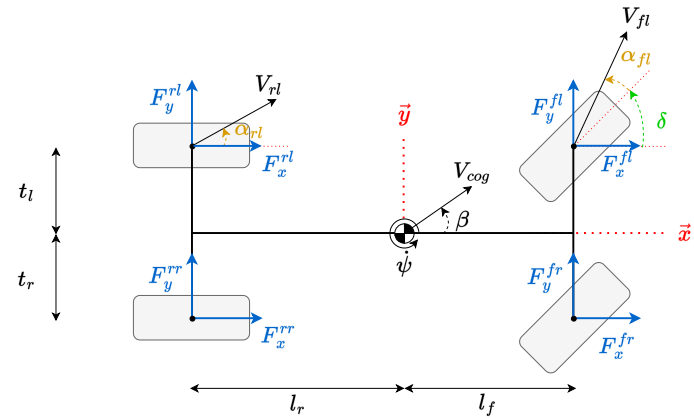


Figure 2.4: Four-wheel vehicle model.

The evolution of the roll, pitch and yaw angular velocities are defined as:

$$
\begin{aligned}
I_x\ddot{\theta} &= -(F_s^{fr}+F_s^{rr})t_r + (F_s^{fl}+F_s^{rl})t_l \\
&\quad +(F_y^{fl}+F_y^{fr}+F_y^{rl}+F_y^{rr})h_{cog} && (2.2a)\\
I_y\ddot{\phi} &= -(F_s^{fl}+F_s^{fr})l_f + (F_s^{rl}+F_s^{rr})l_r - \\
&\quad (F_x^{fl}+F_x^{fr}+F_x^{rl}+F_x^{rr})h_{cog} + (h_{aero}-h_{cog})F_{aero} && (2.2b)\\
I_z\ddot{\psi} &= (F_y^{fl}+F_y^{fr})l_f - (F_y^{rl}+F_y^{rr})l_r + \\
&\quad (F_x^{fr}+F_x^{rr})t_r - (F_x^{fl}+F_x^{rl})t_l && (2.2c)
\end{aligned}
$$

As it is clear from the presented equations, the motion of the vehicle depends on the suspension forces F_s, the vertical forces F_z and the tire forces F_x, F_y. These will be described next.

2.2.4 Suspension Model

The introduced four-wheel model is considered to be equipped with a passive suspension system. A passive suspension is modeled by a spring and a damper which makes the suspension force dependent on the length of the suspension. The suspension force is defined as follows:

$$
\begin{aligned}
F_s^i &= -k_s\Delta z_s^i(\theta,\phi) - d_s(\Delta\dot{z}_s^i(\theta,\phi)) && (2.3)\\
\Delta z_s^i(\theta,\phi) &= \epsilon_i t_i \sin\theta - l_i\cos\theta\sin\phi && (2.4)\\
F_z^i &= F_{z0}^i + F_s^i && (2.5)
\end{aligned}
$$

with i representing the considered wheel (fl, fr, rl, rr), k_s and d_s being the spring and damper stiffness coefficients respectively; $\epsilon_i = -1$ for the left wheels,

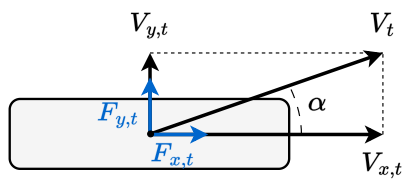


Figure 2.5: Top view of the tire at combined accelerating and cornering.

$\epsilon_i = 1$ for the right wheels; $l_i = l_f$ for the front wheels, $l_i = -l_r$ for the rear wheels; $t_i = t_l$ for the left wheels, $t_i = t_r$ for the right wheels; the vertical forces at rest being $F^i_{z0} = \frac{l_r M_T g}{2(l_f+l_r)}$ for the front wheels and $F^i_{z0} = \frac{l_f M_T g}{2(l_f+l_r)}$ for the rear wheels. The equations are based on explanations provided in [Rajamani 2012] and [Polack 2018].

2.2.5 Tire Model

The equations of motion of the vehicle depend on the tire forces exerted on the vehicle's body. These forces are the result of the controls applied by the driver to the vehicle and are the cause of its movement. The tire-related parameters used in the following analysis are shown in Table 2.3. The tire models presented next describe the wheel forces in the tire frame as shown in Figure 2.5; the transformation of these to the vehicle frame is expressed as follows:

$$F^i_x = (F^i_{xt}\cos\delta_i - F^i_{yt}\sin\delta_i)\cos\phi - F^i_z\sin\phi \tag{2.6}$$

$$\begin{aligned} F^i_y &= (F^i_{xt}\cos\delta_i - F^i_{yt}\sin\delta_i)\sin\theta\sin\phi \\ &\quad +(F^i_{yt}\cos\delta_i + F^i_{xt}\sin\delta_i)\cos\theta + F^i_z\sin\theta\cos\phi \end{aligned} \tag{2.7}$$

While the link between the wheel torques and the tire forces can be expressed as follows:

$$I_r\dot{\omega}_i = T_i - r_{eff}F^i_{xt} \tag{2.8}$$

$\dot{\omega}$ being the angular acceleration of the wheel and T_i being the torque applied at wheel i.

Constants	**Characteristics**	**Unit**
μ	Friction coefficient of the road	-
r_{eff}	Effective tire radius (i.e. distance between the road and the wheel center)	m
I_r	Inertia of the wheel	$\mathrm{kg\,m^2}$
Variables	**Characteristics**	**Unit**
δ	Angle of the front wheel with respect to the axis of the vehicle (steering angle)	rad
τ_i	Longitudinal slip ratio at wheel i	-
α_i	Slip angle at wheel i	rad
V_{xt}^i, V_{yt}^i	Longitudinal and lateral velocities of wheel i	$\mathrm{m\,s^{-1}}$
ω_i	Angular speed of wheel i	$\mathrm{rad\,s^{-1}}$
Forces	**Characteristics**	**Unit**
F_{xt}^i, F_{yt}^i	Longitudinal and lateral forces applied on wheel i expressed in the tire frame	N

Table 2.3: Parameters related to the tire model.

The forces exerted by the road on each tire depend on the following variables:

- τ: The longitudinal slip of the tire expressed as $\tau = \frac{r_{eff}\omega_i - V_{xt,i}}{r_{eff}|\omega_i|}$ when in traction phase and $\tau = \frac{r_{eff}\omega_i - V_{xt,i}}{|V_{xt,i}|}$ when in braking phase.

- α: The slip angle of the tire expressed as $\alpha = \delta - \arctan\left(\frac{V_y + l_f\dot{\psi}}{V_x + \epsilon_i l_w \dot{\psi}}\right)$ for the front wheels and $\alpha = -\arctan\left(\frac{V_y - l_r\dot{\psi}}{V_x + \epsilon_i l_w \dot{\psi}}\right)$ for the rear wheels.

- F_z: The vertical forces introduced in the previous section.

- μ: The friction coefficient of the road.

Modeling the tire forces is linked to many tire models used in the literature. These range from complex tire models as the Pacejka [Pacejka 2006] model, to simple ones as the linear tire model. Each of the models is associated with a validity domain beyond which it is no longer accurate. Different tire models with a decreasing complexity order are presented next.

2.2.5.1 Pacejka Model

The Pacejka tire model also known as the Magic Formula was developed in [Pacejka and Besselink 1997] as a means to describe the behavior of the tire forces with respect to the slip ratio (or slip angle). It is a semi-empirical model.

The general form of the Pacejka model is given by:

$$y = D\sin(C\arctan(Bx - E(Bx - \arctan(Bx)))) \tag{2.9}$$

$$Y = y + S_v \tag{2.10}$$

$$x = X + S_h \tag{2.11}$$

Y represents the output variable: the longitudinal force, the lateral force or the self-aligning torque; and X represents the longitudinal slip ratio (or slip angle). The parameters as defined in [Pacejka 2006] are presented in Table 2.4.

Parameter	Definition	Value
B	Stiffness factor	$\frac{\left(\frac{dy}{dx}\right)_{x=0}}{CD}$
C	Shape factor	$1 \pm (1 - \frac{2}{\pi}\arcsin(\frac{y_s}{D}))$
D	Peak factor	y_{max}
E	Curvature factor	$\frac{Bx_m - \tan(\frac{\pi}{2C})}{Bx_m - \arctan(Bx_m)}$
S_h	Horizontal shift	-
S_v	Vertical shift	-
y_s	Asymptotic value of y	-
x_m	x value corresponding to y_{max}	-

Table 2.4: Parameters of the Pacejka model.

The main parameters of the model, with the force curve with respect to the X-axis are shown in Figure 2.6.

The reference splits the definition into two parts: pure slip and combined slip. In the first case, slips occur separately either longitudinally or laterally while in the second case slips occur simultaneously. For the sake of usability we provide the equations as defined at the reference. The introduced equations describe first the longitudinal and lateral forces and the self-aligning torques for the pure slip case (Equations (2.12)-(2.14)) and then for the combined slip case (Equations

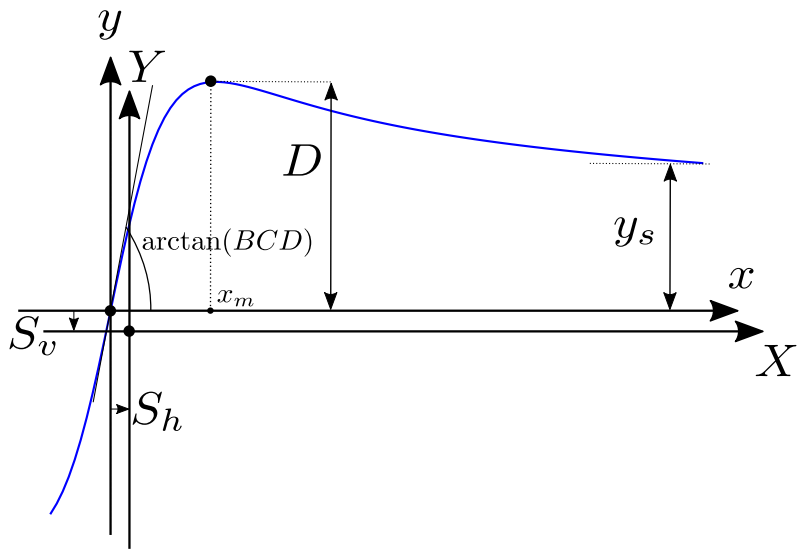


Figure 2.6: Pacejka curve and parameters.

(2.15)-(2.17)). Note that for the combined slip cases, the pure slip longitudinal and lateral forces are multiplied by a factor ($G_{x\alpha}$, $G_{y\tau}$) to represent the actual force.

The scaling factors for each of the cases are defined in Table 2.5. These factors are usually set to unity. Their values may change for situations that involve a change in the friction coefficient or in maneuvers that occur on a wet surface. Note that R_0 is the wheel radius without any load, γ is the camber angle, $\gamma^* = \sin\gamma$ and $\alpha^* = \tan\alpha$. p, q, r and s are wheel related constants.

2.2.5.1.1 Longitudinal Force (pure slip)

$$
\begin{aligned}
F_x &= F_{x0} = D_x \sin[C_x \arctan\{B_x\tau_x - E_x(B_x\tau_x - \\
&\quad \arctan(B_x\tau_x))\}] + S_{Vx} && (2.12a)\\
\tau_x &= \tau + S_{Hx} && (2.12b)\\
C_x &= p_{Cx1}.\lambda_{Cx} && (2.12c)\\
D_x &= \mu_x F_z && (2.12d)\\
\mu_x &= (p_{Dx1} + p_{Dx2} dF_z).\lambda^*_{\mu_x} && (2.12e)\\
E_x &= (p_{Ex1} + p_{Ex2} dF_z + p_{Ex3} dF_z^2)(1 - p_{Ex4}\mathrm{sgn}(\tau_x)).\lambda_{Ex} && (2.12f)\\
K_{x\tau} &= F_z.(p_{Kx1} + p_{Kx2} dF_z).\exp(p_{Kx3} dF_z).\lambda_{K_{x\tau}} && (2.12g)
\end{aligned}
$$

Parameter	Definition
Pure Slip	
λ_{Fz0}	Nominal load
$\lambda_{\mu_{x,y}}$	Peak friction coefficient
$\lambda^*_{\mu_{x,y}}$	Composite friction scaling factor
$\lambda'_{\mu_{x,y}}$	Degressive friction scaling factor
$\lambda_{K_{x\tau}}$	Brake slip stiffness
$\lambda_{K_{y\alpha}}$	Cornering stiffness
$\lambda_{C_{x,y}}$	Shape factor
$\lambda_{E_{x,y}}$	Curvature factor
$\lambda_{H_{x,y}}$	Horizontal shift
$\lambda_{V_{x,y}}$	Vertical shift
$\lambda_{K_{y\gamma}}$	Camber force stiffness
$\lambda_{K_{z\gamma}}$	Camber torque stiffness
λ_{Mr}	Residual torque
Combined Slip	
$\lambda_{x\alpha}$	α influence on F_x
$\lambda_{y\tau}$	τ influence on F_y
$\lambda_{V_y\tau}$	τ induced 'ply-steer' F_y
λ_{My}	Rolling resistance moment
Other	
λ_{Cz}	Radial tire stiffness
λ_{Mx}	Overturning couple stiffness
λ_{My}	Rolling resistance moment

Table 2.5: Scaling factors for the tire forces.

$$B_x = K_{x\tau}/C_x D_x + \epsilon_x \tag{2.12h}$$
$$S_{hx} = (p_{Hx1} + p_{Hx2} dF_z).\lambda_{Hx} \tag{2.12i}$$
$$S_{Vx} = F_z.(p_{Vx1} + p_{Vx2} dF_z).\{|V_{x,t}|/(\epsilon_{V_x} + |V_{x,t}|)|\}.\lambda_{V_x}.\lambda'_{\mu x} \tag{2.12j}$$

2.2.5.1.2 Lateral force (pure slip)

$$F_y = F_{y0} = D_y \sin[C_y \arctan\{B_y\alpha_y - E_y(B_y\alpha_y - \arctan(B_y\alpha_y))\}] + S_{Vy} \tag{2.13a}$$
$$\alpha_y = \alpha^* + S_{Hy} \tag{2.13b}$$
$$C_y = p_{Cy1}.\lambda_{Cy} \tag{2.13c}$$
$$D_y = \mu_y F_z \tag{2.13d}$$
$$\mu_y = \{(p_{Dy1} + p_{Dy2} dF_z)/(1 + p_{Dy3}\gamma^{*2})\}.\lambda^*_{\mu_y} \tag{2.13e}$$
$$E_y = (p_{Ey1} + p_{Ey2} dF_z)\{1 + p_{Ey5}\gamma^{*2} - (p_{Ey3} + p_{Ey4}\gamma^*)\mathrm{sgn}(\alpha_y)\}.\lambda_{Ey} \tag{2.13f}$$
$$K_{y\alpha} = p_{Ky1} F_{z0} \sin[p_{Ky4} \arctan\{F_z/((p_{Ky2} + p_{Ky5}\gamma^{*2} F_{z0}\}]/(1 + p_{Ky3}\gamma^{*2}).\lambda_{Ky\alpha} \tag{2.13g}$$
$$B_y = K_{y\alpha}/(C_y D_y + \epsilon_y) \tag{2.13h}$$
$$S_{Hy} = (p_{Hy1} + p_{Hy2} dF_z).\lambda_{Hy} + (K_{y\gamma 0}\gamma^* - S_{Vy\gamma})/(K_{y\alpha} + \epsilon_K) - 1 \tag{2.13i}$$
$$S_{Vy\gamma} = F_z.(p_{Vy3} + p_{Vy4} dF_z)\gamma^*.\lambda_{Ky\gamma}.\lambda'_{\mu y} \tag{2.13j}$$
$$S_{Vy} = F_z.(p_{Vy1} + p_{Vy2} dF_z).\lambda_{V_y}.\lambda'_{\mu y} + S_{V_y\gamma} \tag{2.13k}$$
$$K_{y\gamma 0} = F_z.(p_{Ky6} + p_{Ky7} F_z).\lambda_{Ky\gamma} \tag{2.13l}$$

2.2.5.1.3 Self-aligning torque (pure side-slip)

$$M_z = M_{z0} = M'_{z0} + M_{zr0} \tag{2.14a}$$
$$M'_{z0} = -t_0.F_{y0} \tag{2.14b}$$
$$t_0 = D_t \cos[C_t \arctan\{B_t\alpha_t - E_t(B_t\alpha_t - \arctan(B_t\alpha_t)\}].\cos\alpha \tag{2.14c}$$
$$\alpha_t = \alpha + S_{Ht} \tag{2.14d}$$
$$S_{Ht} = q_{Hz1} + q_{Hz2} dF_z + (q_{Hz3} + q_{Hz4} dF_z)\gamma^* \tag{2.14e}$$
$$M_{zr0} = D_r \cos[C_r \arctan(B_r\alpha_r)] \tag{2.14f}$$
$$\alpha_r = \alpha + S_{Hf} \tag{2.14g}$$
$$S_{Hf} = S_{Hy} + S_{Vy}/K'_{y\alpha} \tag{2.14h}$$

$$K'_{y\alpha} = K_{y\alpha} + \epsilon_K \tag{2.14i}$$

$$B_t = (q_{Bz1} + q_{Bz2}dF_z + q_{Bz3}dF_z^2). (1 + q_{Bz5}|\gamma^*| + q_{Bz6}\gamma^{*2}).\lambda_{Ky\alpha}/\lambda^*_{\mu_y} \tag{2.14j}$$

$$C_t = q_{Cz1} \tag{2.14k}$$

$$D_{t0} = F_z.(R_0/F_{z0}).(q_{Dz1} + q_{Dz2}dF_z).\lambda_t.\mathrm{sgn}V_{tx} \tag{2.14l}$$

$$D_t = D_{t0}.(1 + q_{Dz3}|\gamma^*| + q_{Dz4}\gamma^{*2}) \tag{2.14m}$$

$$E_t = (q_{Ez1} + q_{Ez2}dF_z + q_{Ez3}dF_z^2) \left\{1 + (q_{Ez4} + q_{Ez5}\gamma^*)\frac{2}{\pi}\arctan(B_tC_t\alpha_t)\right\} \tag{2.14n}$$

$$B_r = (q_{Bz9}.\lambda_{Ky\alpha}/\lambda_{\mu_y} + q_{Bz10}B_yC_y) \tag{2.14o}$$

$$C_r = 1 \tag{2.14p}$$

$$D_r = F_zR_0\{(q_{Dz6} + q_{Dz7}dF_z)\lambda_{Mr} + (q_{Dz8} + q_{Dz9}dF_z)\gamma\lambda_{Kz\gamma} +((q_{Dz10} + q_{Dz11}dF_z)\gamma^*|\gamma^*|)\}\cos\alpha.\lambda^*_{\mu_y}V_{tx} - 1 \tag{2.14q}$$

$$K_{z\alpha 0} = D_{t0}K_{y\alpha\gamma} \tag{2.14r}$$

$$K_{z\gamma 0} = F_zR_0(q_{Dz8} + q_{Dz9}dF_z)\lambda_{Kz\gamma} - D_{t0}K_{y\gamma 0} \tag{2.14s}$$

2.2.5.1.4 Longitudinal Force (combined slip)

$$F_x = G_{x\alpha}.F_{x0} \tag{2.15a}$$

$$G_{x\alpha} = \cos[C_{x\alpha}\arctan\{B_{x\alpha}\alpha_S - E_{x\alpha}(B_{x\alpha}\alpha_S - \arctan(B_{x\alpha}\alpha_S))\}] /G_{x\alpha 0}] \tag{2.15b}$$

$$G_{x\alpha 0} = \cos[C_{x\alpha}\arctan\{B_{x\alpha}S_{Hx\alpha} - E_{x\alpha}(B_{x\alpha}S_{Hx\alpha} - \arctan(B_{x\alpha}S_{Hx\alpha}))\}] \tag{2.15c}$$

$$\alpha_S = \alpha + S_{Hx\alpha} \tag{2.15d}$$

$$B_{x\alpha} = (r_{Bx1} + r_{Bx3}\gamma^{*2})\cos[\arctan(r_{Bx2}\tau)].\lambda_{x\alpha} \tag{2.15e}$$

$$C_{x\alpha} = r_{Cx1} \tag{2.15f}$$

$$E_{x\alpha} = r_{Ex1} + r_{Ex2}dF_z \tag{2.15g}$$

$$S_{Hx\alpha} = r_{Hx1} \tag{2.15h}$$

2.2.5.1.5 Lateral Force (combined slip)

$$F_y = G_{y\tau}.F_{y0} + S_{Vyk} \tag{2.16a}$$

$$G_{y\tau} = \cos[C_{y\tau}\arctan\{B_{y\tau}\tau_S - E_{y\tau}(B_{y\tau}\tau_S - \arctan(B_{y\tau}\tau_S))\}] /G_{y\tau 0}] \tag{2.16b}$$

$$G_{y\tau 0} = \cos[C_{y\tau}\arctan\{B_{y\tau}S_{Hy\tau} - E_{y\tau}(B_{y\tau}S_{Hy\tau} -$$

$$\arctan(B_{y\tau}S_{Hy\tau}))\}] \tag{2.16c}$$

$$\tau_S = \tau + S_{Hy\tau} \tag{2.16d}$$

$$B_{y\tau} = (r_{By1} + r_{By4}\gamma^2)\cos[\arctan\{(r_{By2}(\alpha - r_{By3})\}].\lambda_{y\tau} \tag{2.16e}$$

$$C_{y\tau} = r_{Cy1} \tag{2.16f}$$

$$E_{y\tau} = r_{Ey1} + r_{Ey2}dF_z \tag{2.16g}$$

$$S_{Hy\tau} = r_{Hy1} + r_{Hy2}dF_z \tag{2.16h}$$

$$S_{Vy\tau} = D_{Vy\tau}\sin[r_{Vy5}\arctan(r_{Vy6}\tau)].\lambda_{V_y\tau} \tag{2.16i}$$

$$D_{Vy\tau} = \mu_y F_z.(r_{Vy1} + r_{Vy2}dF_z + r_{Vy3}\gamma).\cos[\arctan(r_{Vy4}\alpha)] \tag{2.16j}$$

2.2.5.1.6 Self-aligning torque (combined slip)

$$M_z = M_z' + M_{zr} + s.F_x \tag{2.17a}$$

$$M_z' = -t.F_y' \tag{2.17b}$$

$$t = D_t\cos[C_t\arctan\{B_t\alpha_{t,eq} - E_t(B_t\alpha_{t,eq} - \arctan(B_t\alpha_{t,eq}))\}]\cos\alpha \tag{2.17c}$$

$$F_y' = F_y - S_{V_y\tau} \tag{2.17d}$$

$$M_{zr} = D_r\cos[C_r\arctan(B_r\alpha_{r,eq})] \tag{2.17e}$$

$$s = R_0.\{s_{sz1} + s_{sz2}(F_y/F_{z0}) + (s_{sz3} + s_{sz4}dF_z)\gamma^*\}\lambda_s \tag{2.17f}$$

$$\alpha_{t,eq} = \sqrt{\alpha_t^2 + \left(\frac{K_{xK}}{Ky\alpha}\right)^2\tau^2}.\mathrm{sgn}(\alpha_t) \tag{2.17g}$$

$$\alpha_{r,eq} = \sqrt{\alpha_r^2 + \left(\frac{K_{xK}}{Ky\alpha}\right)^2\tau^2}.\mathrm{sgn}(\alpha_r) \tag{2.17h}$$

As presented above, the Pacejka tire model requires the knowledge of many parameters to operate successfully. While many of these parameters can be neglected (e.g. the scaling factors shown in Table 2.5), other parameters are essential and may require complex tests to be identified. For this reason, simpler tire models are usually used in the literature at the cost of lower accuracy, especially when high nonlinearities occur. Two widely used simpler models are introduced next. A comparison between the different models is presented later on.

2.2.5.2 Dugoff Model

The Dugoff tire model is an analytical tire model introduced in [Dugoff 1969]. It is not as accurate as the Pacejka model, but it is able to model the tire forces using fewer parameters. It describes the tire forces using Equations (2.18).

$$F_x = C_\tau\frac{\tau}{1-\tau}f(\lambda) \tag{2.18a}$$

$$F_y = C_\alpha \frac{\tan\alpha}{1-\tau} f(\lambda) \tag{2.18b}$$

$$\lambda = \frac{\mu F_z (1+\tau)}{2\sqrt{(C_\tau \tau)^2 + (C_\alpha \tan\alpha)^2}} \tag{2.18c}$$

$$f(\lambda) = \begin{cases} (2-\lambda)\lambda & \text{if } \lambda < 1 \\ 1 & \text{if } \lambda \geq 1 \end{cases} \tag{2.18d}$$

C_τ and C_α being the longitudinal and lateral tire stiffness coefficients respectively; they can be linked to the Pacejka model presented previously as shown in Equations (2.19).

$$C_\tau = B_x C_x D_x \tag{2.19a}$$

$$C_\alpha = B_y C_y D_y \tag{2.19b}$$

Variations of this model have been introduced in the literature (e.g. the modified Dugoff model [Ding and Taheri 2010]) for a more accurate representation of the forces in combined slip scenarios. A simpler model is introduced next.

2.2.5.3 Linear Model

The linear tire model presents a linear relationship between the tire forces and the slip ratio and slip angle. It is valid only for the linear region shown in Figure 2.6. It overestimates the forces for high slips. The model is described in Equations (2.20).

$$F_x = C_\tau \tau \tag{2.20a}$$

$$F_y = C_\alpha \alpha \tag{2.20b}$$

The link to the Pacejka tire model can be done using Equations (2.19). A variation of this model is the linear adaptive tire model that considers the tire stiffness coefficients as variables and not constants; this variation is usually used in state observers as we will present later on.

Remark 3. Other tire models can be found in the literature as the Uni-tire model presented in [Guo and Ren 1999] and the HSRI (Highway Safety Research Institute) model presented in [Tielking and Mital 1974]. Both are semi-empirical models that consider the slips and vertical forces to calculate the tire forces. They are not as popular as the presented above models.

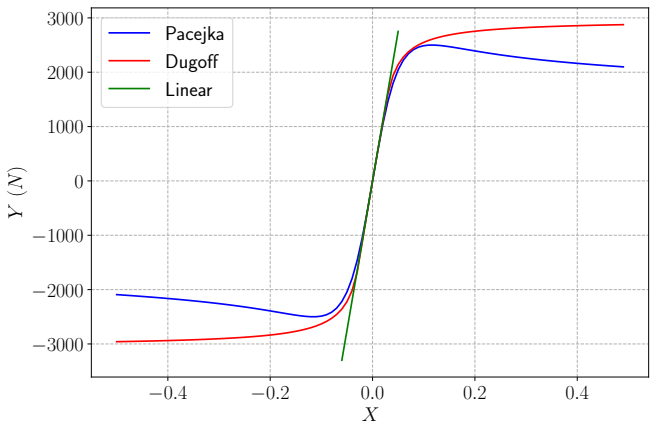


Figure 2.7: Modeling comparison of the different tire models.

2.2.5.4 Discussion

As presented above, a difference exists between the complexity of the described tire models. A comparison between the modeling of the three introduced models is presented in Figure 2.7. The plot assumes $B = 10$, $C = 2.2$, $D = 2500$ and $E = 1$ with $\mu = 1$ and $F_z = 3000\,\text{kN}$; its sole purpose is to illustrate the difference between the computed forces by the different models. Taking the Pacejka model as the reference, the plot shows that the linear model is valid only for the linear region of the tire behavior; while the Dugoff model is able to represent the tire behavior for a wider range even when nonlinearities are available. For high nonlinearities, the Dugoff model presents inaccuracies but is still a better representation than the linear tire model.

Remark 4. The presented four-wheel vehicle model with the Pacejka tire model will be simulated in this thesis using the implementation developed by [Polack 2018].

The presented four-wheel vehicle model used with a Pacejka tire model (as done in [Polack 2018]) could be an accurate representation of the behavior of the vehicle; but it comes with two problems: on the one hand, it necessitates the knowledge of many vehicle and tire parameters: these are not easily accessible; on the other hand, it is computationally heavy and necessitates low integration steps to accurately represent the fast evolving tire dynamics. The mentioned problems affect observers that may lose accuracy when using a model with erroneous parameters, and planners that require a simple model to make fast decisions. For the mentioned reasons, simplified vehicle models are introduced next.

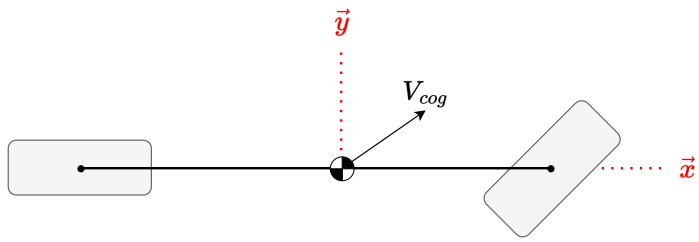


Figure 2.8: The bicycle model.

2.3 Bicycle Model

The bicycle model is a simplification of the previously introduced four-wheel model. It will be described in this section while stating the hypotheses that make it valid. The different vehicle models that use the bicycle model are then detailed: the dynamic bicycle model in Section 2.3.1, the extended bicycle model in Section 2.3.2 and the kinematic bicycle model in Section 2.3.3.

The bicycle model is shown in Figure 2.8. In addition to the previously introduced hypotheses, this model assumes the following:

Hypothesis 3. The four-wheel model can be lumped into a bicycle model; i.e. the two front wheels are represented by a single steerable front wheel and the two rear wheels are represented by a single non-steerable rear wheel.

Hypothesis 4. The vehicle is moving on a road with a slope angle Θ_s and a road bank angle $\Theta_b = 0$.

Hypothesis 5. The pitch, roll and vertical dynamics are neglected.

The different bicycle models are defined next.

2.3.1 Dynamic Bicycle model

2.3.1.1 Description

The dynamic bicycle model studies the dynamics of the vehicle. As shown in Figure 2.9, this model assumes that the forces of the two front wheels are equal, and can be modeled by the longitudinal F_x^f and lateral F_y^f front wheel forces and that the forces of the two rear wheels are equal and can be modeled by the longitudinal F_x^r and lateral F_y^r rear wheel forces. The remaining parameters presented on the figure are the ones used in the four-wheel model introduced in Table 2.2.

The state of the system consists of its longitudinal V_x and lateral V_y velocities and its yaw rate $\dot{\psi}$. The equations governing the state evolution of the system are presented next.

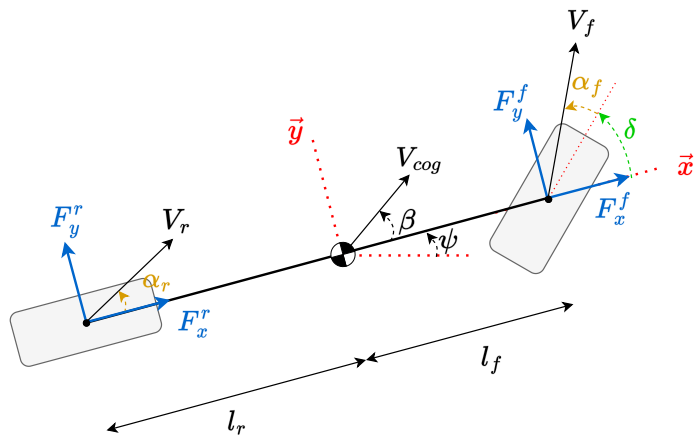


Figure 2.9: The dynamic bicycle model.

2.3.1.2 Physics of the System

The evolution of the state of the system is described in Equations (2.21), derived using the second law of Newton.

$$M_T\dot{V}_x = M_T\dot{\psi}V_y + F_x^f + F_x^r - F_{aero} - M_T g \sin\Theta_s \tag{2.21a}$$
$$M_T\dot{V}_y = -M_T\dot{\psi}V_x + F_y^f + F_y^r \tag{2.21b}$$
$$I_z\ddot{\psi} = F_y^f l_f - F_y^r l_r \tag{2.21c}$$

The control inputs can be either the longitudinal and lateral wheel forces; or the longitudinal acceleration $a_x = \dot{V}_x - \dot{\psi}V_y$ and the steering angle δ then is integrated in the tire forces equations. The wheel forces are dependent on the wheel model used. The wheel model used depends on the design choice; the wheel models presented in Section 2.2.5 can be employed.

Note that the evolution of the position of the center of gravity in the inertial frame is described by:

$$\dot{X} = V_x \cos\psi - V_y \sin\psi \tag{2.22a}$$
$$\dot{Y} = V_x \sin\psi + V_y \cos\psi \tag{2.22b}$$

The dynamic bicycle model is able to accurately represent the behavior of the vehicle given its hypotheses are valid. Though, it is dependent on a tire model which makes its accuracy related to the validity of the tire model chosen.

A non-dynamic bicycle model is introduced next.

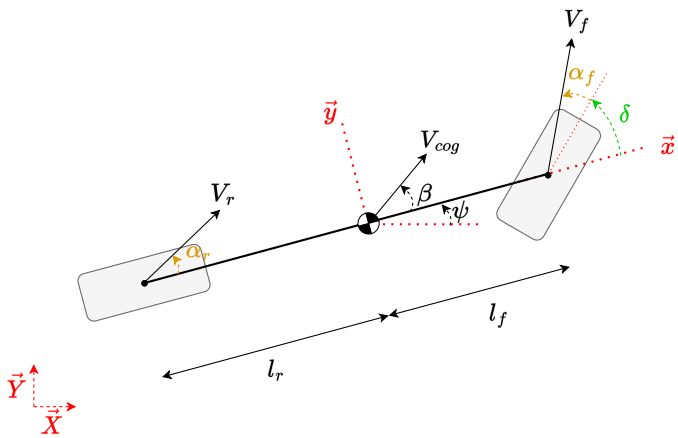


Figure 2.10: The extended bicycle model.

Remark 5. The dynamic bicycle model can be further simplified by representing the evolution of the lateral dynamics only with a state consisting of the side-slip angle β and the yaw rate $\dot{\psi}$ while considering a linear tire model [Choi et al. 2002].

2.3.2 Extended Bicycle Model

2.3.2.1 Description

The extended bicycle model shown in Figure 2.10 does not take into account the dynamics of the system. It is concerned with the kinematics of the vehicle while including the slip angles at the front and rear wheels. The kinematics study of a system is concerned with its motion without any reference to the forces or masses entailed in it. This model was presented in [Lenain et al. 2003] as an augmentation of the kinematic bicycle model that will be presented in Section 2.3.3 to account for slips in the vehicle's motion. In addition to the previous hypotheses, this model assumes the following:

Hypothesis 6. The vehicle is moving on a road with a slope angle $\Theta_s = 0$ and a road bank angle $\Theta_b = 0$.

The state of the system consists of its X, Y coordinates and its heading ψ. The equations governing the state evolution of the system, described by [Spentzas et al.; Lenain et al. 2001; 2003] are presented next.

2.3.2.2 Physics of the System

The state evolution of the extended bicycle model is expressed in the inertial frame shown in Figure 2.1. The coordinates of the center of gravity shown in Figure

2.10 are expressed as (X_{cog}, Y_{cog}). The state evolution of the system is described in Equations (2.23).

$$\dot{X}_{cog} = V_{cog} \cos(\beta + \psi) \tag{2.23a}$$
$$\dot{Y}_{cog} = V_{cog} \sin(\beta + \psi) \tag{2.23b}$$
$$\dot{\psi} = \frac{V_{cog} \cos\beta(\tan(\delta + \alpha_f) + \tan\alpha_r)}{l_f + l_r} \tag{2.23c}$$
$$\beta = \arctan\left(\frac{-l_f \tan\alpha_r + l_r \tan(\delta + \alpha_f)}{l_f + l_r}\right) \tag{2.23d}$$

The control inputs are the velocity V_{cog} and the steering angle δ.

Although this model introduces accurate properties through the inclusion of the slip angles at the wheels, the difficulty remains in identifying the slip angles without referring to the vehicle's dynamics.

A simpler bicycle model is introduced next.

2.3.3 Kinematic Bicycle Model

2.3.3.1 Description

The kinematic bicycle model shown in Figure 2.11 is a kinematic representation of the motion of the vehicle as the extended model presented previously. In addition to the properties of the previous model, the kinematic bicycle model assumes the following:

Hypothesis 7. No slip is present at the wheels.

This makes the kinematic bicycle model only valid for low-speed scenarios.

Similarly to the extended bicycle model, the state of the kinematic bicycle model consists as well of its X, Y coordinates and its heading ψ. The equations governing the state evolution of the system are presented next.

2.3.3.2 Physics of the System

The state evolution of the kinematic bicycle model is expressed in the inertial frame shown in Figure 2.1. The coordinates of the center of gravity shown in Figure 2.11 are expressed as (X_{cog}, Y_{cog}). The state evolution of the system is described in Equations (2.24).

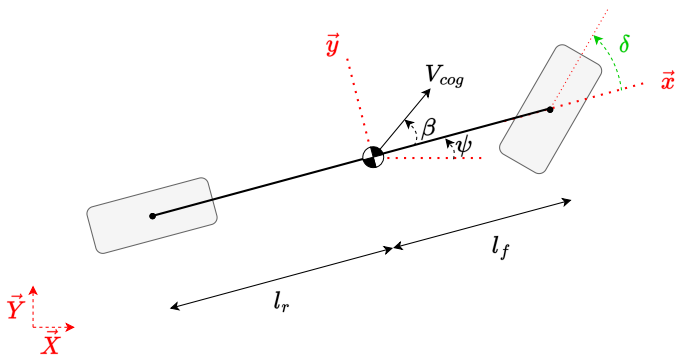


Figure 2.11: The kinematic bicycle model.

$$\dot{X}_{cog} = V_{cog} \cos(\beta + \psi) \tag{2.24a}$$

$$\dot{Y}_{cog} = V_{cog} \sin(\beta + \psi) \tag{2.24b}$$

$$\dot{\psi} = \frac{V_{cog} \cos\beta \tan\delta}{l_f + l_r} \tag{2.24c}$$

$$\beta = \arctan\left(\frac{l_r \tan\delta}{l_f + l_r}\right) \tag{2.24d}$$

The control inputs are the velocity V_{cog} and the steering angle δ.

The advantage of the kinematic bicycle model lies in it being a simple model that requires very few parameters to model the motion of the vehicle. It was shown in [Polack et al. 2017] to be a valid model as long as the lateral acceleration condition $a_y < 0.5\mu g$ is satisfied.

Having presented the three bicycle models, we move next to a simpler model that has less constraints, the point mass model.

2.4 Point Mass Model

The point mass model is the simplest representation of the motion of the vehicle that we introduce in this document. In what follows, the point mass model is described and the equations governing its state evolution are derived.

2.4.1 Description

The point mass model represents the vehicle as a single point mass at its center of gravity able to move in all directions as shown in Figure 2.12. The derivation of

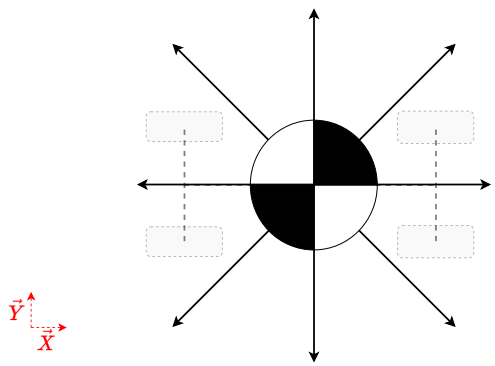


Figure 2.12: The point mass model.

the point mass model follows, in addition to the already introduced hypotheses, the following:

Hypothesis 8. The model is reduced to two degrees of freedom.

Hypothesis 9. The yaw ψ, roll θ, pitch ϕ angles are neglected.

The state of the system is defined by its coordinates (X, Y) in the inertial frame and their derivatives $(\dot{X}, \dot{Y})$.

2.4.2 Physics of the System

Following the above-stated hypotheses, the state evolution of the system is defined by Equations (2.25).

$$\ddot{X} = a_X \tag{2.25a}$$
$$\ddot{Y} = a_Y \tag{2.25b}$$

The control inputs are the longitudinal and lateral accelerations a_X and a_Y in the inertial frame.

As the point mass model is able to move in all directions, variations to it were introduced in the literature (e.g. [Godbole et al.; Altché et al. 1997; 2017]) to make it more representative of the motion of the vehicle through well-defined constraints. Though, the point mass model relies on many hypotheses which makes its representation of the behavior of the vehicle uncertain.

2.5 Conclusion

In this chapter, we have presented vehicle and tire models with different complexities. We started by presenting the four-wheel vehicle model to which we added

hypotheses that led to simplified vehicle models. Different tire models were introduced as well.

In brief, none of the introduced models is considered to be ideal to model the behavior of the vehicle. The choice of the model used depends on the situation the vehicle is in and the maneuver it is trying to achieve. Using a simple model requires less parameter knowledge and results in higher computational efficiency at the expense of losing accuracy in dynamic scenarios while using a complex model leads to more accurate behavior representation but requires more parameter knowledge and results in lower computational efficiency.

As mentioned before, vehicle models are used as the basis of model-based observers and planners, and the accuracy of these will depend on the accuracy of the vehicle model used; this will be investigated in the upcoming chapters.

The next chapter will make use of the presented models to design model-based state observers. The accuracy of the used models will be questioned when comparisons will be made to the approaches developed in this thesis.

Furthermore, the introduced models will be employed in model predictive planners later-on, to which the performance of the approach developed in this thesis will be compared.

Part II

State Observers

Chapter 3

Model-based State Observers

Contents

3.1 Introduction **38**

3.2 Observability **39**

3.3 Types of Observers **39**

3.3.1 Luenberger Observer 40

3.3.2 Extended Luenberger Observer 41

3.3.3 Sliding Mode Observer 42

3.3.4 Kalman Filter 43

3.3.5 Extended Kalman Filter 44

3.4 Application to Vehicles **45**

3.4.1 Side-slip angle observers 47

3.4.2 Velocity Observers 49

3.4.3 Full State Observers 50

3.4.4 Discussion 51

3.5 Model Validity in Observers **51**

3.5.1 EKF definition 52

3.5.2 Testing data generation 53

3.5.3 EKF testing 54

3.6 Conclusion **55**

3.1 Introduction

The knowledge of a vehicle's states is key to successful autonomous driving applications. As stated in chapter 1, planning trajectories and controlling the vehicle based on inaccurate state knowledge results in inaccurate vehicle behavior. While many of the state variables of a vehicle can be measured using in-car conventional sensors, the accurate access to non-measurable and noisy measured states requires the use of state observers.

State observers are employed either for the reconstruction of the state of the system (i.e. accessing the non-measurable states); to filter the noisy measures; or to fuse redundant noisy measures to reach accurate estimations (i.e. sensor fusion). As shown in Figure 3.1, observers take as inputs the controls applied to the system and the measures generated by the sensors to deliver the state observations.

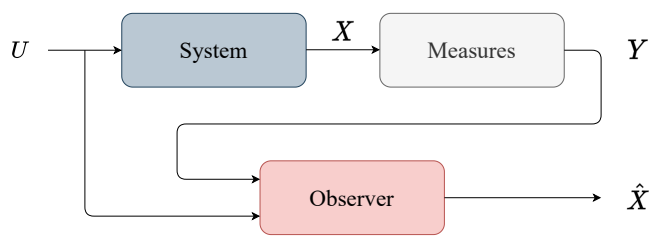


Figure 3.1: Block diagram of a state observer. U denotes the control inputs, X denotes the state of the system, Y denotes the measures and $\hat{X}$ denotes the predicted state by the observer.

State observers can be either model-based, depending on a model representing the state evolution of the system to deliver state observations; or learning-based, depending on neural networks to deliver state observations. This chapter will focus on model-based observers, while learning-based observers will be targeted in the next chapters.

Model-based observers should be able to account for process noise and measurement noise. Process noise is defined as the difference of behavior present between the actual system and its model which is caused by the assumptions underlying the model and by external disturbances. While measurement noise is defined as the difference between the measurements of the actual system and its actual state.

Model-based observers make use of the vehicle models presented in the previous chapter to model the state evolution of the vehicle. They introduce a gain term to account for the differences between the modeled variables and the measured ones, aiming for accurate state observations. Several model-based observing algorithms are introduced in the literature with varying complexity and consid-

erations: they will be presented in this chapter.

This chapter will start by introducing the observability concept which is a requisite to the functioning of observers; then will introduce the different observing algorithms ranging from simple linear observers to more complex ones. Their applications to the vehicles in the literature will be highlighted.

3.2 Observability

A system is observable if it is possible to determine its state from the observation of its measurable output over a finite time interval. Observability is a concept introduced by Kalman [Kalman 1960a] and is an important one because it determines the existence of a solution to the observing problem. The Kalman observability criteria for a given system is defined below.
Consider a linear, time-invariant system defined by:

$$\dot{X} = AX + BU \tag{3.1a}$$

$$Y = CX \tag{3.1b}$$

with X being the state of the system of shape n, U being the input to the system, and Y being the measurement vector. The system is said to be observable if the observability matrix shown in Equation (3.2) is of rank n.

$$O = \begin{bmatrix} C \\ \cdots \\ CA \\ \cdots \\ \vdots \\ \cdots \\ CA^{n-1} \end{bmatrix} \tag{3.2}$$

The use of state observers in a system requires the system to be observable. If a system is not observable, it is not possible to infer its states from its outputs.

Next, observing algorithms presented in the literature will be introduced, then their use in vehicle models will be highlighted.

3.3 Types of Observers

Different types of observers are presented in the literature to observe the state of a system. In all of the presented observers, the algorithm will take into consideration the model describing the state evolution of the system and the available measurements to output its estimations. The complexity of the observing algorithms can range from the linear Luenberger observer presented in Section 3.3.1

to the Kalman Filter presented in Section 3.3.4 and its variations. Each of the observers will be detailed next.

3.3.1 Luenberger Observer

The Luenberger observer, known also as a state observer, is a linear estimator that uses the linear state space representation of a system to predict its state with a correction term that includes the difference between the predicted observation and the actual measurement. A block diagram of the Luenberger observer is shown in Figure 3.2 describing the equations presented next.

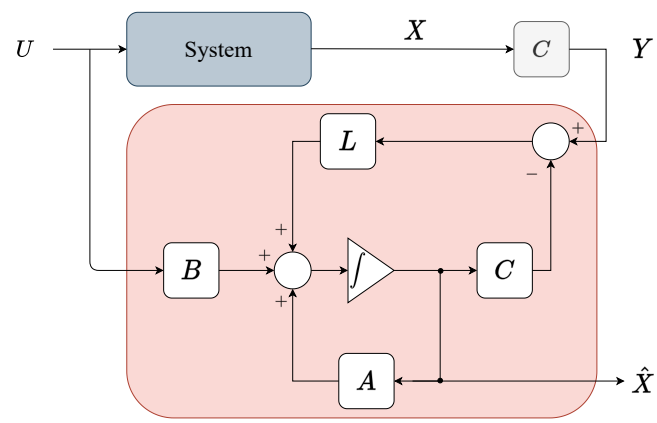


Figure 3.2: Luenberger observer block diagram.

Consider the system shown below:

$$\dot{X} = AX + BU \tag{3.3a}$$
$$Y = CX \tag{3.3b}$$

with X being the state of the system, U being the input to the system, and Y being the measurement vector. The observed state of the system is defined as:

$$\dot{\hat{X}} = A\hat{X} + BU + L(Y - C\hat{X}) \tag{3.4}$$

with $\hat{X}$ being the estimated state and L being the observer gain. Thus, the error between the predicted state and the actual state is:

$$e = X - \hat{X} \tag{3.5}$$

and the error dynamics are determined by:

$$\dot{e} = \dot{X} - \dot{\hat{X}} = (A - LC)e \tag{3.6}$$

The convergence of the observer is determined by the eigenvalues of matrix $A-LC$. The gain L can be, for example, determined by pole placement. As many control systems are nonlinear, the Luenberger observer is extended to a nonlinear form presented next.

3.3.2 Extended Luenberger Observer

The Extended Luenberger observer is introduced to observe nonlinear systems. The same logic used on the linear observer is used to construct a nonlinear one. A block diagram of this observer is shown in Figure 3.3 describing the equations presented next.

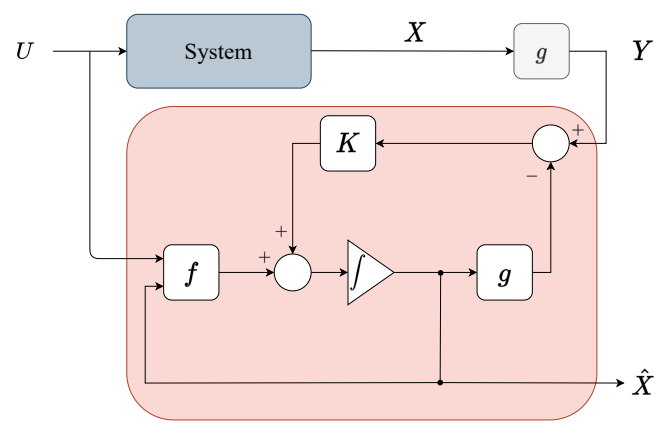


Figure 3.3: Extended Luenberger observer block diagram.

Consider the nonlinear system shown below:

$$\dot{X} = f(X, U) \tag{3.7a}$$
$$Y = g(X) \tag{3.7b}$$

with X being the state of the system, U being the input to the system, and Y being the measurement vector. $f(X, U)$ is the function describing the evolution of the system, and $g(X)$ is the function describing the output of the system. A structure similar to the one used in linear observers is proposed:

$$\dot{\hat{X}} = f(\hat{X}, U) + K(Y - \hat{Y}) \tag{3.8}$$

with $\hat{X}$ being the predicted state and K being the observer gain. K should make the observation converge to the system state. The error between the predicted state and the actual state then is:

$$e = X - \hat{X} \tag{3.9}$$

and the error dynamics are determined by:

$$\dot{e} = \dot{X} - \dot{\hat{X}} = f(X, U) - f(\hat{X}, U) + K(g(\hat{X}) - g(X)) \tag{3.10}$$

The challenging part relies in proving the convergence of a nonlinear observer. The Lyapunov direct method is usually used to determine the gain term K to be used based on a Lyapunov function $V(\hat{X}, X)$ that reflects the error of the observer.

The Lyapunov direct method states that if in a neighborhood of the equilibrium point X_0 exists a positive definite function $V(X)$ with continuous first derivatives and $\dot{V}(X)$ is negative semi-definite then X_0 is stable for the nonlinear system. If $\dot{V}(X)$ is negative definite then X_0 is asymptotically stable.

After introducing the nonlinear observer, the sliding mode observer is introduced next.

3.3.3 Sliding Mode Observer

Sliding mode observing is a nonlinear observing method that feeds back the estimation to the predictor as a nonlinear switching term instead of feeding it back as a linear term as done in the Luenberger observer mentioned previously. The concept of sliding mode observers appeared in the late 1980s [Canudas De Wit and Slotine 1989] in a work for robot manipulators and was further improved after. The block diagram of the sliding mode observer is shown in Figure 3.4.

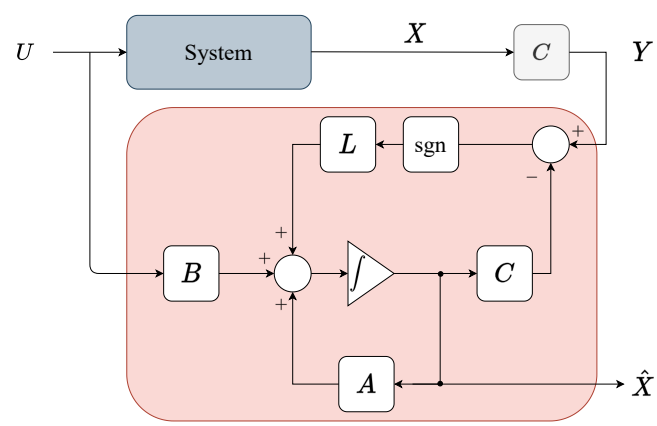


Figure 3.4: Sliding mode observer block diagram.

The sliding mode observer introduced in [Drakunov and Utkin 1995] considers the following system:

$$\dot{X} = AX + BU \tag{3.11a}$$

$$Y = CX \tag{3.11b}$$

with X being the state of the system, U being the input to the system, and Y being the measurement vector. A sliding mode observer replaces the correction term of the regular state observer by a discontinuous term. The prediction becomes as follows:

$$\dot{\hat{X}} = A\hat{X} + BU + L.\text{sgn}(Y - C\hat{X}) \tag{3.12}$$

The sign of the switching function $s = Y - C\hat{X}$ multiplies L. The goal is, with a proper choice of L, to reach the sliding surface $Y - C\hat{X} = 0$. Sliding mode observers can be applied to linear and nonlinear systems and are usually combined with Luenberger observers. Proving the convergence of these observers can be done using the Lyapunov direct method discussed in the previous section.

So far, presented observers deal with the system to be observed with no regard to the properties of the present process or measurement noise. The observers detailed next, make use of the above presented algorithms while taking into account the process and measurement noise of the system.

3.3.4 Kalman Filter

In the following, we consider a linear discrete system subject to process and measurement noise. The considered system will be observed using the Kalman filter presented in this Section. The system is defined by:

$$X_k = F_k X_{k-1} + B_k U_k + w_k \tag{3.13a}$$

$$Z_k = H_k X_k + v_k \tag{3.13b}$$

k being the current time step, X_k being the state of the system at k, Z_k being the measurement vector at k, F_k being the state transition model, B_k being the control input model, and H_k being the observation model. Noises are assumed to be Gaussian: w_k being the process noise with covariance Q_k and v_k being the measurement noise with covariance R_k.

The Kalman filter [Kalman 1960b] is a linear optimal filter. It deals with a linear system subject to process and measurement noises as presented above. The Kalman filter algorithm is split into two steps: the predict step and the update step. In the predict step, the algorithm will predict the current state and covariance based on the previous state and covariance and the current applied inputs, while in the update step, it will correct the predicted state and covariance based on the current measurements. The equations defining the Kalman filter are the following:

Predict

$$\hat{X}_{k|k-1} = F_k \hat{X}_{k-1|k-1} + B_k U_k \tag{3.14a}$$

$$P_{k|k-1} = F_k \hat{P}_{k-1|k-1} F_k^T + Q_k \tag{3.14b}$$

Update

$$\tilde{Z}_k = Z_k - H_k \hat{X}_{k|k-1} \tag{3.14c}$$

$$S_k = H_k P_{k|k-1} H_k^T + R_k \tag{3.14d}$$

$$K_k = P_{k|k-1} H_k^T S_k^{-1} \tag{3.14e}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \tilde{Z}_k \tag{3.14f}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1} \tag{3.14g}$$

With $\hat{X}_{k|k-1}$ being the predicted state estimate, $P_{k|k-1}$ being the predicted covariance estimate, $\tilde{Z}_k$ being the measurement residual, S_k being the covariance residual, K_k being the Kalman gain, $\hat{X}_{k|k}$ being the updated state estimate, and $P_{k|k}$ being the updated covariance estimate.

As seen in the presented Equations, the structure of the Kalman filter follows the structure of the above presented observers, computing a state estimate and adding to it a gain term K that multiplies the difference between the measurements and the computed state ($\hat{Z}_k$). The difference being that the optimal Kalman gain K takes into consideration the covariance estimate and the covariance residual.

As the Kalman filter deals with linear systems, it is extended next to nonlinear applications.

3.3.5 Extended Kalman Filter

The Extended Kalman Filter (EKF) is a nonlinear version of the Kalman filter introduced to deal with nonlinear systems by linearizing at each step around the current estimate.
Consider the following nonlinear system subject to both process noise and measurement noise:

$$X_k = f(X_{k-1}, U_k) + w_k \tag{3.15a}$$

$$Z_k = g(X_k) + v_k \tag{3.15b}$$

Noises are assumed to be Gaussian: w_k being the process noise with covariance Q_k and v_k being the measurement noise with covariance R_k.
The EKF algorithm has the same structure of the Kalman filter algorithm.The equations of the EKF are the following:

Predict

$$\hat{X}_{k|k-1} = f(\hat{X}_{k-1|k-1}, U_k) \tag{3.16a}$$

$$P_{k|k-1} = F_k \hat{P}_{k-1|k-1} F_k^T + Q_k \tag{3.16b}$$

Update

$$\tilde{Z}_k = Z_k - g(\hat{X}_{k|k-1}) \tag{3.16c}$$

$$S_k = G_k P_{k|k-1} G_k^T + R_k \tag{3.16d}$$

$$K_k = P_{k|k-1} G_k^T S_k^{-1} \tag{3.16e}$$

$$\hat{X}_{k|k} = \hat{X}_{k|k-1} + K_k \tilde{Z}_k \tag{3.16f}$$

$$P_{k|k} = (I - K_k G_k) P_{k|k-1} \tag{3.16g}$$

F_k is the state transition matrix and G_k is the observation matrix defined as:

$$F_k = \left.\frac{\partial f}{\partial X}\right|_{\hat{X}_{k-1|k-1}, U_k} \tag{3.17a}$$

$$G_k = \left.\frac{\partial G}{\partial X}\right|_{\hat{X}_{k-1|k-1}} \tag{3.17b}$$

As opposed to the Kalman filter, the EKF is not an optimal filter. Though, it is widely used in the literature for observing applications. Other variations to the Kalman filter were introduced in the literature, e.g. the Unscented Kalman Filter [Wan and Van Der Merwe 2000], the Invariant Kalman Filter [Bonnabel 2007].

It should be noted that although the Kalman filter and its variations present more accurate observations due to the inclusion of process and measurement noise properties, the knowledge of the Q and R matrices are a necessity to their functioning.

We have presented above different types of observing algorithms with the description of the properties of each of them. Other model-based observing algorithms could be present in the literature however we presented the ones that we noticed were the most used for vehicle applications.

In what follows, we describe the use of the presented algorithms to vehicle applications to observe different quantities.

3.4 Application to Vehicles

An accurate knowledge of the state of the vehicle is essential to its functioning: planning trajectories or controlling the vehicle with no accurate knowledge about its state will result in inaccurate behavior. For this purpose, this topic was the focus of many research works that used observers to estimate either one or many states of the vehicle.

In what follows, we present the work done in the literature for the estimation of vehicle states. We divide the literature into three categories:

- Side-slip angle observers.
- Velocities (V_x, V_y) and yaw rate ($\dot{\psi}$) observers.
- Full vehicle state observers.

Each of the categories is represented in a Table (Tables 3.1, 3.2, 3.3) showing the type of observer used and the vehicle model and tire model employed inside the used observer.

We notice that the different presented observers, in each of the categories, rely on different techniques to accomplish the estimation objective. The differences appear in the model used and the considered state to be estimated, in the available sensor measurements, and in the observer's algorithm. This will be discussed in each of the categories presented next.

Reference	**Vehicle model used**	**Tire model used**	**Observer used**
[Lee et al. 2013]	Dynamic/Kinematic bicycle model	Linear model	Luenberger
[Kiencke and Daiß 1997]	Dynamic bicycle model	Linear tire	Luenberger
[Cherouat et al. 2005]	Dynamic bicycle model	Linear tire	Luenberger
[Piyabongkarn et al. 2009]	Dynamic/Kinematic bicycle model	Linear tire	Luenberger
[Song et al. 2014]	Dynamic bicycle model	Dugoff model	Extended Luenberger
[Hac and Simpson 2000]	Dynamic bicycle model	-	Extended Luenberger
[Liu et al. 2012]	Four-wheel model	-	Sliding mode
[Baffet et al. 2007]	Dynamic bicycle model	Adaptive linear model	Sliding mode and EKF
[Chen and Hsieh 2008]	Dynamic/Kinematic bicycle model	Linear model	EKF
[Reina and Messina 2019]	Dynamic bicycle model	Adaptive linear model	EKF
[Gao and Yu 2010]	Dynamic bicycle model	Pacejka model	EKF
[van Aalst et al. 2018]	Dynamic bicycle model	Adaptive linear model	EKF
[Hrgetic et al. 2014]	Four-wheel model	-	EKF
[Wang et al. 2010]	Dynamic bicycle model	Modified Dugoff model	-

Table 3.1: Model-based side-slip angle observers applied to vehicles.

Reference	**Vehicle model used**	**Tire model used**	**Observer used**
[Yin et al. 2017]	Dynamic bicycle model	-	Extended Luenberger
[Zhao et al. 2011]	Four-wheel model	Dugoff model	Extended Luenberger
[Ma et al. 2017]	Four-wheel model	Uni-tire model	Extended Luenberger
[Zhao Linhui et al. 2008]	Dynamic bicycle model	Dugoff model	Extended Luenberger
[Guo et al. 2016]	Four-wheel model	-	Extended Luenberger and Sliding mode
[Zhao et al. 2009]	Four-wheel model	Dugoff model	Sliding mode
[Kim et al. 2018]	Dynamic bicycle model	-	EKF

Table 3.2: Model-based velocities V_x, V_y and yaw rate $\dot{\psi}$ observers applied to vehicles.

Reference	Vehicle model used	Tire model used	Observer used
[Sebsadji et al. 2008]	Four-wheel model	Dugoff model	Luenberger and EKF
[Wenzel et al. 2006]	Four-wheel model	Pacejka model	EKF
[Zong et al. 2013]	Four-wheel model	HSRI	EKF

Table 3.3: Model-based full vehicle state observers applied to vehicles.

Remark 6. Approaches in the presented tables with no tire model specification either employ an independent tire forces observer connected to the main observer, or include the tire forces in the state of the system to be observed.

Remark 7. The approach that does not show an observer algorithm specification integrates the measurements in the update equations but do not include a state correction term.

Remark 8. In the presented real-vehicle literature applications, both the inputs U and the measurements Y are provided by sensor measurements.

3.4.1 Side-slip angle observers

As seen in chapter 2, the side-slip angle of a vehicle is the angle between the velocity vector and its longitudinal axis; it is a fundamental quantity to assess vehicle stability especially for critical driving situations. Moreover, it is part of the state in many non-linear models of vehicle dynamics. Though, the side-slip angle is only measurable with expensive sensors, either based on optical flow directly over ground or highly accurate dual-antenna GNSS (Global Navigation Satellite System) solutions. For these reasons, the side-slip angle is a typical application for state observers in the literature.

Different methods to observe the side-slip angle of a vehicle are shown in Table 3.1. The table presents the different vehicle models, tire models and observing algorithms employed to reach the goal.

3.4.1.1 Methodology

Being the angle between the velocity vector at the center of gravity and the longitudinal axis of the vehicle, the side-slip angle can be described as:

$$\beta = \arctan \frac{V_y}{V_x} \tag{3.18}$$

Which relates it to the longitudinal and lateral vehicle dynamics of the four-wheel and dynamic bicycle model used at the basis of the presented observers. A kinematic version of the side-slip angle is shown in Equation (2.24d) and is used in the kinematic based observers.

Differences between works appear in the model state definition, the available measurements, and the observing algorithm.

The state of the model includes in some applications, in addition to the side-slip angle, other states or parameters from which the side-slip angle estimation can be inferred. The additional states used depend on both the vehicle and tire model at the base of the observer. For example, a dynamic bicycle model can be described by a state containing the side-slip angle and the yaw rate of the vehicle (e.g. [Lee et al.; Reina and Messina 2013; 2019]) or by a state containing the longitudinal and lateral velocities and the yaw rate, from which the side-slip angle can be calculated (e.g. [Liu et al.; van Aalst et al. 2012; 2018]). The designer may decide to include additional variables as the cornering stiffness (e.g. [Baffet et al. 2007]) or the tire forces (e.g. [Song et al.; Baffet et al. 2014; 2007]) expanding the state to be estimated.

The measurements differ as well between applications. Most of the applications assume that the steering angle, the accelerations and the yaw rate are measurable: these can be linked directly to the dynamics evolution of the vehicle, others assume as well access to the wheel speeds (e.g. [Song et al.; Hrgetic et al. 2014; 2014]) or even the longitudinal velocities (e.g. [Hrgetic et al. 2014]) .

The used state and measurements are processed by an observing algorithm which choice differs between applications as seen in the table.

The unusual observer designs encountered for the side-slip angle estimation are summarized in the following points:

- Many of the works implementing a linear tire model tend to present the cornering stiffnesses as variables which results in a linear adaptive model aiming to reduce tire dynamics errors (e.g. [Lee et al.; Reina and Messina 2013; 2019]).

- The work in [Song et al. 2014] presented a method to calculate the peak force based on torque sensors, feeding it to the Dugoff tire model for more accurate results.

- The work in [Baffet et al. 2007] introduced two types of observers interacting together: a sliding mode observer that feeds the estimations of the forces and the yaw rate to an EKF that will output the side-slip angle estimations.

- The work in [van Aalst et al. 2018] introduced a virtual measurement of the side-slip angle based on a bicycle model with an adaptive linear tire model, to be fed to an EKF resulting in the final side-slip angle estimation based on the estimated longitudinal and lateral velocities.

Next, we discuss velocity observers.

3.4.2 Velocity Observers

As seen in the vehicle modeling chapter 2, the longitudinal and lateral velocities and yaw rate are involved in the state of the dynamic vehicle models. Thus, their accurate knowledge leads to the identification of the state of the vehicle. On the one hand, the longitudinal and lateral velocities are not measurable using standard car sensors and access to them necessitates the use of state observers; on the other hand, the yaw rate can be accessed easily, but its filtering is the target of many vehicle observers.

The different methods used in the literature to observe the velocities and yaw rate of the vehicle are shown in Table 3.2. In contrast to the side-slip angle table, we remark that only the dynamic bicycle model and the four-wheel vehicle model are at the base of these observers.

3.4.2.1 Methodology

The estimation of the vehicle's longitudinal and lateral velocities and yaw rate is based on the four-wheel model or the dynamic bicycle model presented in chapter 2, more precisely on Equations (2.1) and (2.21). These include the evolution of the velocities and yaw rate based on the mass of the vehicle and the forces applied to it.

The differences between the used observers to estimate the needed quantities are, as stated above, at the level of the models used, the state to be estimated, the measurements used, and the observer algorithm.

Two main models are used in the considered literature: the four-wheel model and the dynamic bicycle model, which makes the state to be estimated made of the longitudinal and lateral velocities and the yaw rate in most of the cases. Several works make the addition of the road slope or road bank angles (e.g. [Zhao et al.; Ma et al.; Kim et al. 2009; 2017; 2018]). While some works include the tire forces in the state of the system to be observed (e.g. [Guo et al.; Kim et al. 2016; 2018]).

When it comes to measurements, common used quantities are the steering angle, the accelerations and the yaw rate, these quantities are easily accessible through standard sensors. Some works include the torque measurements from which the longitudinal tire forces are inferred using Equation (2.8) (e.g. [Zhao Linhui et al.; Guo et al.; Yin et al. 2008; 2016; 2017]).

The choice of the observer used varies between the different applications as seen in the table. Applications in [Guo et al. 2016] and [Kim et al. 2018] use cascaded, connected observers: in these works the estimation is split into two steps, a first observer estimates the states required for the functioning of the second observer; the second observer estimates the needed quantities using the estimations of the first observer.

Full state observers are presented next.

3.4.3 Full State Observers

While the previously presented observers focus on a specific state or number of states (e.g. velocities), several observers try to achieve a full state estimation. Note that the state of a vehicle has no fixed definition and is dependent on the vehicle model used and the designer choice. For the different considered full state estimation works shown in Table 3.3, we will detail the choices made and the methodology used next.

3.4.3.1 Methodology

To be able to account for multiple state variables in the observation of the vehicle, an accurate model is used. This can be seen in the model section of the corresponding table showing that a four-wheel vehicle model was the choice of all the considered works. This is reflected as well in the tire model choice that does not include any linear or adaptive linear implementations.

As mentioned in the previous sections, the differences of the models, the considered state, the measurements and observing algorithms are present between the presented works.

The work in [Sebsadji et al. 2008] considers the variables to be estimated to be the tire slip ratio and angles, the longitudinal and lateral velocities, the side slip angle and the slope angle of the road. While in the works [Wenzel et al. 2006] and [Zong et al. 2013], two state vectors are introduced, one consisting of the system's variables and the other consisting of the system's parameters: the two vectors differ based on the assumptions of each application. In [Wenzel et al. 2006], the variable state vector includes the velocities and yaw rate, the accelerations, the slip angles and ratios of the wheels, the side slip angle, the vertical forces and the roll angle; the parameter vector includes the mass of the vehicle, its inertia about the z axis and the length between the position of the center of gravity with respect to the front wheels. While in [Zong et al. 2013], the variable state vector includes the same variables of [Wenzel et al. 2006] except for the roll angle, while the parameter vector consists of the road friction coefficients μ at each wheel.

When it comes to the measurements, the works in [Sebsadji et al. 2008] and [Wenzel et al. 2006] assume access to the steering angle, wheel speeds, accelerations and yaw rate; while in [Zong et al. 2013], the accelerations and the yaw rate are the only quantities used.

All of the mentioned works consider a two-step observation. In [Sebsadji et al. 2008] both steps (EKF then Luenberger) aim for a final full state observation;

whether in [Wenzel et al. 2006] and [Zong et al. 2013], one step observes the constant parameters while the other one observes the variable state vector.

3.4.4 Discussion

Having presented different model-based observing strategies targeting different vehicle variables based on the vehicle and tire models introduced in chapter 2, their limitations are discussed.

The performance of the presented observers faces three main challenges:

- First, their accuracy is linked to the validity of the hypotheses associated with the model they use. Observing the state of a vehicle with a nonvalid model will lead to inaccuracies (e.g. Observing the state of a vehicle involved in high dynamic scenarios using a dynamic bicycle with a linear tire model, knowing that the linear tire model is only valid for low dynamic scenarios).

- Second, the use of more detailed models (such as four-wheel vehicle models, or Pacejka tire models) requires the knowledge of more vehicle and tire parameters; the improper knowledge of these parameters may affect the accuracy of the observer.

- Third, the measurements used are often noisy, affecting the accuracy of the observer depending on them.

The model validity case is further demonstrated in a simulated observing application next.

All of the three cases are presented on real vehicle applications in Chapter 5, in comparison to the approach developed later on in this thesis.

In addition to the issues associated with the used model, the choice of the observing algorithm impacts the estimation accuracy. Using an observer that takes into account the process and measurement noise (Kalman filter) will deliver better results than a simple state observer (Luenberger) given that the properties of the process and measurement noise are known.

3.5 Model Validity in Observers

Having presented the different observer applications and deduced the model validity issues that govern them, we implement a simple simulated observing approach to investigate the presented claims.

The implemented observing application aims to observe the coordinates X, Y and the heading ψ of a four-wheel vehicle model (with a Pacejka tire model) using two EKFs based on two simplified vehicle models: the kinematic bicycle model (KBM) and the dynamic bicycle model (DBM) with a linear tire model. All of

the used models are based on the explanations provided in Chapter 2. The goal is to test the ability of the simplified models to observe the mentioned state of the four-wheel vehicle model with the defiance of their hypotheses. The observers will be tested on lane change trajectories with various speeds and their performance will be compared.

3.5.1 EKF definition

As stated earlier, the goal of the performed tests in this section is to conclude the effect of model validity in observing the state of the vehicle through observing the state of the four-wheel model with EKFs based on simplified models. To achieve this purpose two EKFs are considered:

- A kinematic bicycle based EKF
- A dynamic bicycle based EKF (with a linear tire model)

To implement these observers, we consider a discrete system inside the implemented EKFs defined by:

$$z_{k+1} = z_k + f(z_k, u_k).\Delta t + w_k \tag{3.19a}$$
$$m_k = g(z_k) + v_k \tag{3.19b}$$

where k is the time step, $\Delta t = 0.01\,\mathrm{s}$ is the time interval, m_k is the measurement, w_k is the process noise and v_k is the measurement noise.

The function $f(z_k, u_k)$ refers to Equations (2.24) of the kinematic bicycle for the first EKF and to Equations (2.21) of the dynamic bicycle model for the second EKF with the neglection of the road slope angle and the aerodynamic forces, with z being the state of the considered model and u being the controls of the model. For the kinematic model, $z = \begin{bmatrix} X & Y & \psi \end{bmatrix}^\top$ and $u = \begin{bmatrix} V & \delta \end{bmatrix}^\top$ while for the dynamic model, $z = \begin{bmatrix} X & V_x & Y & V_y & \psi & \dot{\psi} \end{bmatrix}^\top$ and $u = \begin{bmatrix} a_x & \delta \end{bmatrix}^\top$. The measurements are assumed to be the coordinates of the vehicle and its heading, $m = \begin{bmatrix} X_m & Y_m & \psi_m \end{bmatrix}^\top$ with X_m, Y_m and ψ_m being the measured variables. In simulation terms, these are derived by adding measurement noise to the simulated vehicle state as defined in the next section.

The EKFs are then implemented given the Equations presented in Section 3.3.5.

Note that the process noise is not added explicitly as it represents the modeling differences between the considered simplified models and the four-wheel model.

The procedure for generating the testing data is defined next.

3.5.2 Testing data generation

As the aim is to observe the coordinates X, Y and the heading ψ of a four-wheel vehicle model, this model should be used to generate the needed testing data. The testing trajectory is chosen to be a single lane change maneuver (inspired by the the ISO-3888-1 standard[1]) effected at varying speeds ranging between $5\,\mathrm{m\,s^{-1}}$ and $17.5\,\mathrm{m\,s^{-1}}$ with 2.5 increments resulting in six trajectories. The reference path is shown in Figure 3.5.

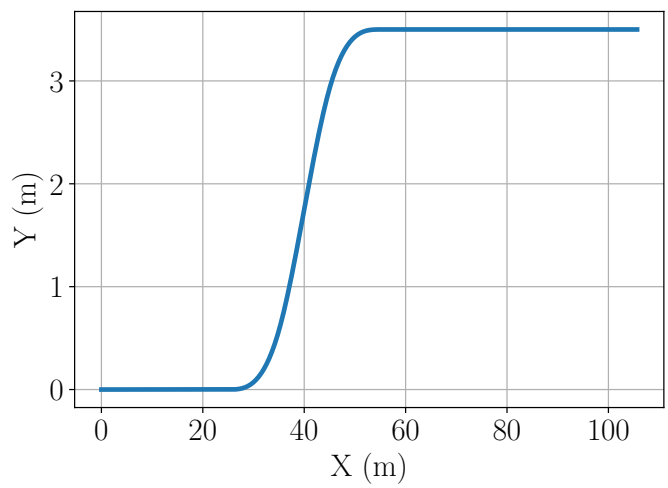


Figure 3.5: Single lane change maneuver used for testing the observers.

Remark 9. The used lateral controller was not able to reach an accurate behavior at $20\,\mathrm{m\,s^{-1}}$ which explains the limitation of the test set to $17.5\,\mathrm{m\,s^{-1}}$.

To be able to generate the required data sets, the vehicle should be controlled to follow the given path at different speeds. Two separate longitudinal and lateral control strategies were used for this purpose. The longitudinal controller aims to make the four-wheel model stick to its target speed using a simple proportional controller that controls the torque T of the model given the error between the actual model speed and the target speed. The lateral controller aims to make the four-wheel model stick to its reference path using a Stanley controller [Thrun et al. 2006] that controls the steering angle δ of the model. The Stanley controller was implemented by Stanford University. It controls the steering angle of the vehicle based on the lateral and heading errors between the reference path and the front axle of the vehicle.

[1] https://www.iso.org/standard/67973.html

After generating the six lane change trajectories, measurement noise is added to the simulated X, Y and ψ variables. The considered measurement noise is white Gaussian with standard deviations being $\sigma_X = 1\,\mathrm{m}$, $\sigma_Y = 1\,\mathrm{m}$ and $\sigma_\psi = 17.4\,\mathrm{mrad}$.

Having the generated trajectories, the previously presented EKF observers are tested next.

3.5.3 EKF testing

The EKFs defined above are tested on the six lane change trajectories. At the start of the observing process, the initial state of each of the EKFs is initiated to the initial state of the four-wheel model. The process noise covariance is modified according to the simplified model accuracy with respect to the reference model in each trajectory while the measurement noise covariance is constant based on the added noise in the previous section. The evaluation metric used below is the absolute error between the ground truth values simulated by the four-wheel model and the estimations of each of the observers.

For each of the lane change trajectories, both EKFs are simulated and their mean and max absolute errors are calculated. The testing results of each lane change trajectory along the speed and the maximum lateral acceleration of the trajectory are shown in Table 3.4. The table shows that both observers are able to provide accurate observations for the low dynamic trajectories with an advantage for the dynamic bicycle based EKF, which is expected given that the parameters provided for its car-body and tire models are the same as the simulated ones in the four-wheel model.

V	$a_y^{\max}$	**EKF model**	**Mean AE** X	**Max AE** X	**Mean AE** Y	**Max AE** Y	**Mean AE** ψ	**Max AE** ψ
5	0.23	KBM	0.004	0.015	0.04	0.17	2.25	9.5
5	0.23	DBM	0.004	0.013	0.013	0.027	0.49	2.1
7.5	0.49	KBM	0.008	0.042	0.081	0.37	5.15	21.9
7.5	0.49	DBM	0.017	0.056	0.015	0.042	1.0	2.9
10	1.08	KBM	0.034	0.13	0.14	0.66	8.06	32.0
10	1.08	DBM	0.012	0.03	0.023	0.061	1.5	7.1
12.5	2.22	KBM	0.047	0.18	0.19	0.77	11.2	46.4
12.5	2.22	DBM	0.018	0.062	0.041	0.10	2.4	8.3
15	4.76	KBM	0.15	0.54	0.32	1.50	12.1	49.4
15	4.76	DBM	0.07	0.24	0.146	0.79	6.2	24.2
17.5	11.27	KBM	0.49	1.92	0.75	2.91	14	48.8
17.5	11.27	DBM	0.17	0.70	0.75	2.92	13.9	48.2

Table 3.4: EKF tests for the different lane change trajectories showing the observing error increase with the harshness of the maneuver. V in $\mathrm{m\,s^{-1}}$, a_y in $\mathrm{m\,s^{-2}}$, X and Y errors in m, ψ errors in mrad. AE: absolute error.

Remark 10. The outperformance of the KBM-based EKF to the DBM-based one

in the second scenario is due to assuming low process noise in the DBM-based EKF leading to the accumulation of the process errors. This is not the case in the following scenarios.

Though, it was expected that both models would lose accuracy when approaching the $a_y = 0.5\,\mathrm{g}$ limit as discussed in [Mitschke; Polack et al. 1990; 2017]. The effects of this claim can be seen in the errors of the corresponding EKFs. Both EKFs present higher errors as the lateral acceleration increases, with an advantage to the dynamic model up to the $a_y^{\max} = 4.76\,\mathrm{m\,s^{-2}}$ scenario; beyond this, both models present high observation errors (that are close for the lateral Y, ψ estimations) as seen for the last trajectory that includes a very harsh maneuver depicted by $a_y^{\max} = 11.27\,\mathrm{m\,s^{-2}}$.

In brief, the presented tests show the performance degradation associated with model validity issues of model-based observers. As this experiment targets a simulated model, additional model-based observers will be implemented later on in real applications and their performance will be compared to that of the proposed approaches in this thesis. A more developed testing procedure will be then presented.

3.6 Conclusion

In this chapter, we presented the different model-based observing algorithms present in the literature and then discussed their use in vehicle state estimation.

Although all of the presented observers are dependent on a vehicle model, the way they integrate the model in their observing procedure differs between algorithms. The Luenberger, Extended Luenberger and Sliding Mode observers do not account for process and measurement noise as it is the case for the Kalman and Extended Kalman filters. We established that the accuracy of the observers depends on the observing algorithm and the validity of the vehicle and tire models used in the observer.

In brief, model-based observers are popular in the literature as they present accurate performance for low dynamic applications; their accuracy is challenged in nonlinear, high dynamic scenarios where the representation of the vehicle behavior becomes more difficult. To solve issues associated to the model validity and the observing algorithm in vehicle observers, we will target learned observers next. We aim to increase the accuracy of state observers by reducing their dependency on vehicle models and deterministic observing algorithms using neural networks.

Chapter 4

Learning-based Observers

Contents

4.1 Introduction **58**

4.2 Neural networks **58**

4.2.1 Overview 58

4.2.2 Stochastic Gradient Descent 59

4.2.3 Feedforward Neural Networks 61

4.2.4 Convolutional Neural Networks 62

4.2.5 Recurrent Neural Networks 63

4.2.6 Transformers 66

4.3 Learned Observers **67**

4.4 CNN-based observer **68**

4.4.1 Considered system 69

4.4.2 Data generation 70

4.4.3 Observer architecture 72

4.4.4 Results and analysis 74

4.5 Conclusion **79**

4.1 Introduction

In the previous chapter, we established the importance of state observers in autonomous driving applications. We explored model-based state observers and discussed their limitations. In this chapter, we introduce the concept of learning to state observers to solve the problems associated with model-based observers. The aim of a learned observer will be to observe the state of the vehicle using the control inputs and measurements.

While machine learning is a vast field, this thesis will only focus on neural networks trained in a supervised manner. Supervised learning requires each training data point to contain input features and an associated label; in observing terms, it requires the ground truth data to which the output of a neural network based observer will be compared. Ground truth data is easily accessible in simulated environments, but difficulties will arise in real environments. This will be discussed further as we reach real vehicle applications.

Using learning is motivated by the ability of neural networks to learn complex nonlinear functions. In our case, we seek learning the system dynamics and measurement noise properties to the neural network which is expected to provide accurate estimations.

This chapter will start by introducing supervised learning methods while stating the applications of each of them; the introduced methods used in learned observers in the literature will be then presented. The chapter will end by applying a learning method to motivate the use of learned observers, which will constitute a preliminary work for the application of the learned observers to real vehicles in the next chapter.

The explanations provided in this chapter are based on [Goodfellow et al. 2016] unless stated otherwise.

4.2 Neural networks

4.2.1 Overview

Artificial neural networks, usually called neural networks, are computational models motivated by the structure and functioning of biological neural networks. They are designed to approximate relationships between input and output data using interconnected artificial neurons.

The early works on neural networks were simple linear models motivated by neuroscience. Their role was to link a set of inputs $x_1, ..., x_n$ to an output y based on a set of weights $w_1, ..., w_n$ through a function $y = f(x, w) = x_1 w_1 + ... + x_n w_n$. In a preliminary work in 1943 [McCulloch and Pitts 1943] the weights would be set manually, but in a more developed work in the 1950s [Rosenblatt 1958] the

weights could be learned using an algorithm considered as a special case of the stochastic gradient descent which is used in training neural networks today.

The advancements in neural networks increased slowly over the years with the promotion of the back-propagation algorithm in the 1980s [Rumelhart et al.; Le Cun and Fogelman-Soulié 1986; 1987], while in the 1990s advancements related to sequences modeling were made [Bengio et al.; Hochreiter and Schmidhuber 1994; 1997].

Although advancements were present, neural networks were difficult to train, due to their computational cost and the hardware limitations at the time. Starting 2006 [Hinton et al. 2006], fast growth in the neural network field began and was influenced by new training strategies and higher computational capabilities. At the current time, neural networks are used in a large range of applications across domains, such as computer vision, natural language processing, speech and audio processing, bio-medicine, autonomous vehicles, financial applications and many other fields.

In this thesis, we aim to use neural networks for observing and planning applications in autonomous vehicles. For this purpose, we start by defining the stochastic gradient descent method used to train neural networks, then we define several types of neural networks used in the literature or employed in our applications later on.

4.2.2 Stochastic Gradient Descent

Training neural networks aims to find the set of weights able to minimize a predefined loss function. Weights are numerical values assigned to the connections between neurons representing their contribution.

Suppose we have a function $y = f(x)$ that we seek to minimize, the derivative of this function is denoted as $f'(x)$ and gives the slope of $f(x)$ at x. Assuming that ϵ denotes a small change in the input, Taylor's first-order approximation leads to $f(x + \epsilon) \approx f(x) + \epsilon f'(x)$. In other words, $f'(x)$ specifies how to slightly alter the input to get a corresponding change in the output. Thus, the derivative is useful to minimize the given function $f(x)$. This is called the gradient descent technique. When dealing with multiple inputs, partial derivatives are used to indicate the change of f with respect to each input x^i at point x, the gradient of f denoted $\nabla_x f(x)$ contains the mentioned partial derivatives.

When it comes to neural networks optimization, the loss function to be minimized, by adjusting the network's weights, is denoted as $L(x, y, W)$, x denoting the inputs to the network, y denoting the outputs of the network and W denoting the weights of the network. The gradient descent algorithm requires the

computation of the gradient:

$$g = \frac{1}{m} \sum_{i=1}^{m} \nabla_W L(x^i, y^i, W) \tag{4.1}$$

m denoting the size of the training set.

It is clear that the computational cost of this operation will increase drastically with the increase of the size of the training set. For this reason the stochastic gradient descent algorithm, based on the gradient descent is introduced.

The stochastic gradient descent algorithm takes a mini-batch of samples drawn uniformly from the training set to compute an estimate of the gradient. The size of the mini-batch is chosen to be small (between one and a few hundred samples) and the estimate of the gradient is denoted as:

$$g = \frac{1}{m'} \nabla_W \sum_{i=1}^{m'} L(x^i, y^i, W) \tag{4.2}$$

with m' being the size of the mini-batch. This concept allows the training of networks using large datasets.

The algorithm to perform learning requires the computation of the gradient that is usually done using the backpropagation [Rumelhart et al. 1986] algorithm which uses the chain rule to propagate the loss function from the output of the network all the way to its input.

The described procedure uses a training dataset to complete the weights optimization of the network.

Before moving to the different types of neural networks, several important points should be noted:

- A training dataset is used to optimize the weights of the network which are initialized using different techniques. A common weight initialization technique is the Xavier initialization [Glorot and Bengio 2010] which consists of assigning weights generated using a uniform distribution between $[-a, a]$, $a = \sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}$, n_{in} being the number of input units to the weight tensor and n_{out} being the number of output units.

- A validation dataset is used when training the model to assess its performance on unseen data. It helps to avoid overfitting the weights of the model to the training set. It has no impact on the learning process but is used to know when to stop training the network. This technique is referred to as early stopping.

- A testing dataset, different than the training and validation sets, assesses the performance of the network after being trained.

- Neural network architectures contain a set of hyperparameters that should be tuned to result in the best possible performance, these include the number of layers, the number of neurons per layer (defined in the next sections) in addition to the learning rate (step size at each iteration), batch size (number of training samples per iteration) and other parameters. A common technique for hyperparameter tuning is grid search [Bergstra et al. 2011]. Grid search relies on testing combinations of several parameters before choosing the ones that result in minimal loss. It should be noted that grid search could be computationally expensive especially when dealing with a high number of parameters or with wide ranges. Other techniques like random search [Bergstra and Bengio 2012] or Bayesian optimization [Snoek et al. 2012] could be used.

Different types of neural networks are defined next.

4.2.3 Feedforward Neural Networks

Feedforward neural networks, also known as multilayer perceptrons are a type of neural networks where the flow of computations starts from the input x through the network to the final output y without any feedback loops (as it is the case for recurrent neural networks presented later-on). An example of feedforward neural networks is shown in Figure 4.1 showing two inputs forming the input layer, a hidden layer made of 4 neurons and an output layer. Activation functions are usually implemented at each layer. Activation functions are non-linear functions applied to introduce nonlinearity to the outputs of chosen layers.

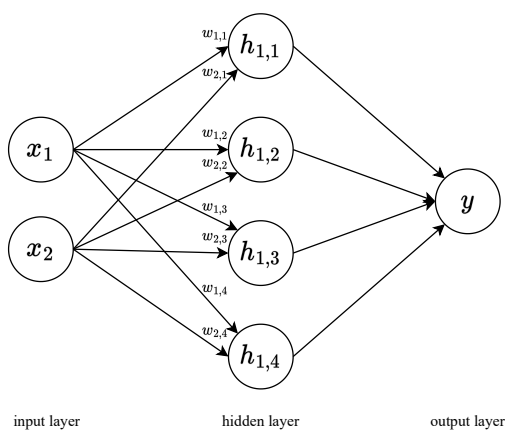


Figure 4.1: Multilayer Perceptron example.

In general, the equations describing the behavior of the neural network are

the following:

$$h^0 = x \tag{4.3a}$$
$$h^k = \sigma^k(W^{(k)\top} h^{(k-1)} + b^k) \text{ for } k = 1..L \tag{4.3b}$$

with x being the input vector, $h^{(k)}$ being the output of layer k, σ^k being the activation function of the k-th layer, W^k being the k-th weight vector and b being the bias. L is the number of layers and h^L is the output layer ($y = h^L$).

Remark 11. In the remaining parts of this thesis, the terms feedforward networks, fully connected layers, multilayer perceptions, dense layers, and linear layers are used interchangeably.

Next, we introduce another type of neural networks.

4.2.4 Convolutional Neural Networks

Convolutional neural networks (CNNs) [LeCun et al. 1989] are another type of neural networks that deals with grid-like data (e.g. images, time series). They make use of convolutions instead of matrix multiplications (as opposed to the previously introduced feedforward networks) in at least one of their layers. It should be noted that the flow of computations in CNNs starts from the input to the output with no feedback loops.

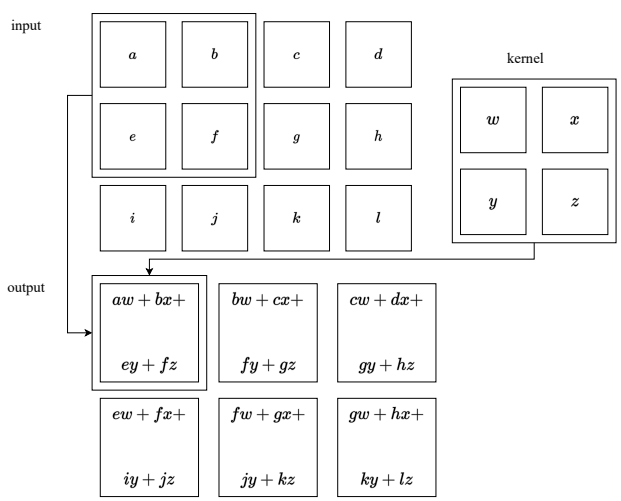


Figure 4.2: Example of 2D convolution as shown in [Goodfellow et al. 2016].

The network emphasizes on weight sharing by using a single kernel to output a feature map. This is shown in Figure 4.2 where the same kernel convolves different parts of the input grid to produce the output feature map.

Another concept used in CNNs is pooling. Pooling is a technique used to reduce the spatial dimension of a feature map while preserving essential information. The most popular pooling operation is the max pooling [Zhou and Chellappa 1988] which selects the maximum value within each region. Pooling is used to make the network more robust to small translations and rotations in the input. The downsampling reduces the computational complexity of the following layers as well.

Typical CNN architectures include a set of convolution and pooling layers before feeding the last feature map to a multilayer perceptron.

In general, the equations describing the behavior of the network are the following:

$$h^0 = x \tag{4.4a}$$

$$h^k = \sigma^k(\pi^{(k)}(W^{(k)\top} * h^{(k-1)} + b^k)) \text{ for } k = 1..L \tag{4.4b}$$

with x being the input vector, $h^{(k)}$ being the output of layer k, σ^k being the activation function of the k-th layer, $\pi^{(k)}$ being the k-th pooling function, W^k being the k-th weight vector and b being the bias. L is the number of layers and h^L is the output layer ($y = h^L$).

Recurrent neural networks are introduced next.

4.2.5 Recurrent Neural Networks

4.2.5.1 Definition

Recurrent neural networks (RNNs) are a type of neural networks able to recognize dependencies in sequences of data. As their name indicates, recurrent neural networks take into account information from subsequent time steps, relying on temporal dependencies to compute the output. As seen in Figure 4.3, the information to be kept between time steps is passed through the hidden states of the network.

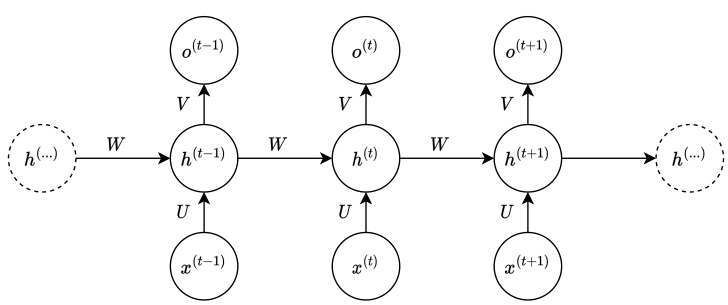


Figure 4.3: A recurrent neural network block.

A single RNN cell (assuming a hyperbolic tangent activation function) is described by the following equations:

$$a^{(t)} = b + Wh^{(t-1)} + Ux(t) \tag{4.5a}$$
$$h^{(t)} = \tanh\left(a^{(t)}\right) \tag{4.5b}$$
$$o^{(t)} = c + Vh^{(t)} \tag{4.5c}$$

with $x^{(t)}$ being the input vector at t, $h^{(t)}$ being the hidden state at t, $o^{(t)}$ being the output at t, U, V and W being the weight matrices and b and c being the bias vectors.

RNNs are trained using back propagation through time (BPTT), which back-propagates through long input sequences. This may lead to a common problem known as the vanishing (or exploding) gradient [Bengio et al. 1994]. Vanishing (or exploding) gradient refers to the gradient significant decrease (or increase) while being back propagated to almost vanish (or saturate) in the early layers which creates difficulties in improving the cost function. To solve this problem, Long Short-Term Memory networks were introduced; they are presented next.

4.2.5.2 Long Short-Term Memory

As mentioned previously, the functioning of recurrent neural networks is restricted by the vanishing (or exploding) gradient problem. To solve this problem, [Hochreiter and Schmidhuber 1997] introduced the Long Short-Term Memory (LSTM) neural networks.

The introduced neural network architecture makes use of several concepts:

- Memory cell: The memory cell, also known as the cell state, serves as a long-term memory able to capture information over long sequences. It is able to retain information across time steps.

- Forget gate: The forget gate controls the amount of information to be discarded from the cell state. It makes use of the previous hidden state and the current input to output a value between 0 and 1 for each element of the cell state determining which information is no longer relevant.

- Input gate: The input gate decides how much new information should be added to the cell state. It makes use of the previous hidden state and the current input to output a value between 0 and 1 determining which values should be updated in the cell state.

- Cell state update: The cell state is updated taking into consideration the outputs of each of the forget and input gates. Operations are performed element-wise to selectively update the cell state.

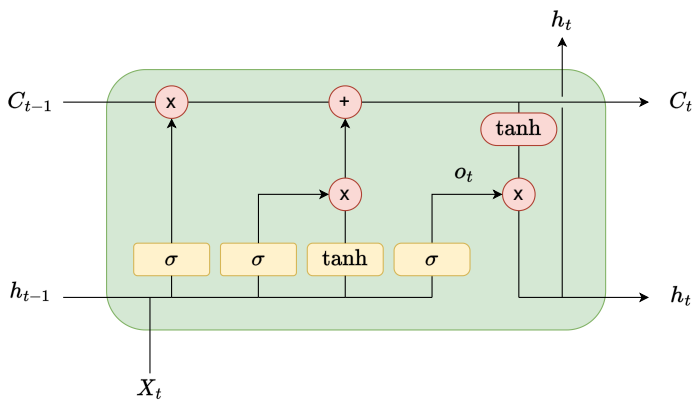


Figure 4.4: LSTM Cell [Hochreiter and Schmidhuber 1997].

- Output gate: The output gate controls the flow of information from the cell state to the output/hidden state.

An LSTM cell diagram is shown in Figure 4.4 and the equations describing its functioning are the following:

$$i_t = \sigma(W_i[h_{t-1}, X_t] + b_i) \tag{4.6a}$$
$$f_t = \sigma(W_f[h_{t-1}, X_t] + b_f) \tag{4.6b}$$
$$o_t = \sigma(W_o[h_{t-1}, X_t] + b_o) \tag{4.6c}$$
$$\tilde{C}_t = \tanh(W_c[h_{t-1}, X_t] + b_c) \tag{4.6d}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{4.6e}$$
$$h_t = o_t \odot \tanh(C_t) \tag{4.6f}$$

where X_t is the input vector, i_t is the input gate, f_t is the forget gate, o_t is the output gate, $\tilde{C}$ is the potential cell state, C is the cell state, h_{t-1} is the previous hidden state vector and h_t is the output vector; σ represents a sigmoid function; W_i, W_f W_o, W_c are the weights and b_i, b_f b_o, b_c are the biases. $\odot$ represents the element-wise product.

Several variants of the LSTM networks were introduced in the literature; as well, other architectures were put in place to solve the problems of traditional RNNs, e.g. Gated Recurrent Units (GRU) [Chung et al. 2014] with similar properties to LSTM but with fewer parameters (no output gate).

While recurrent networks have been useful when dealing with input sequences, transformers have emerged recently as an alternative to RNNs gaining more popularity especially in natural language processing (NLP) approaches. They are briefly discussed next.

4.2.6 Transformers

As opposed to recurrent neural network architectures, transformers rely on a self-attention mechanism as presented in the "Attention is all you need" work [Vaswani et al. 2017]. In NLP applications, the self-attention mechanism allows transformers to consider the context of a word or token by simultaneously processing all other words in the sequence, rather than sequentially as in RNNs and LSTMs, making them more efficient and suitable for handling long sequences.

The self-attention mechanism computes a correlation score between words or tokens in a sequence representing the strength of their relationship through a dot product multiplication. This correlation score is used to assign weights to the links between words or tokens in the sequence. The weights are updated through the learning process.

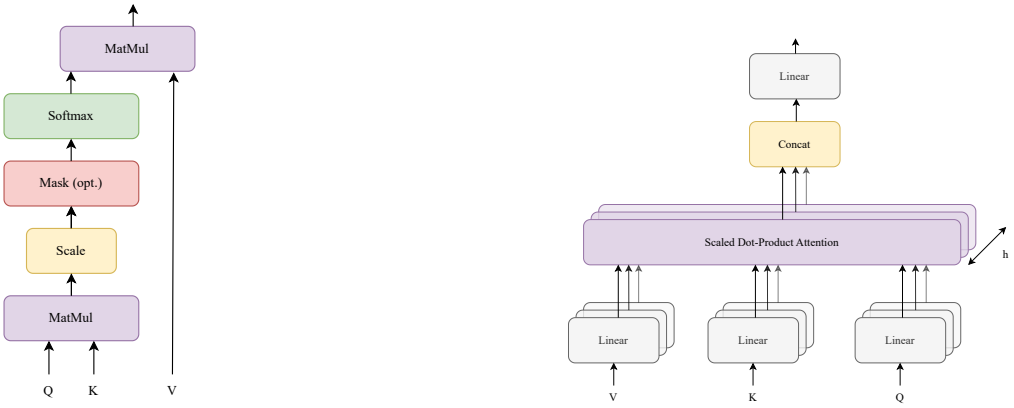
The computation of the self-attention weights in the attention mechanism relies on queries, keys and values, denoted by three learnable weight matrices Q, K and V applied to the same input. The attention function maps a query and a set of key-value pairs to an output. The dimension of the queries and keys being d_k and the dimension of the values being d_v. The weights on the values are obtained by computing the dot products of the query with all keys (to get the correlation matrix), dividing each by $\sqrt{d_k}$ (to normalize the values) and applying a softmax function (to get the attention weights, i.e. the correlation probabilities). Multiplication by the value V matrix results in the attention output. The values corresponding to tokens with higher attention weights will have a more significant impact on the final output. The scaled dot-product attention is then defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4.7}$$

In the mentioned work, the authors use multi-head attention, linearly projecting the queries, keys and values, h times with learned linear projections. The attention function is then applied to each of the projected versions yielding output values. These are projected again resulting in the final values. Both the scaled dot-product attention and the multi-head attention are illustrated in Figure 4.5 as presented in the reference.

Additionally, transformers employ positional encoding to incorporate the order of the words or tokens in the sequence. This allows the model to differentiate between words with the same content but different positions, enabling it to capture the sequential dependencies in the data.

A transformer architecture involves incorporating positional encoding into the inputs of the network, which are then passed through a series of multi-head attention and feedforward layers. The structure presented in the reference makes use of an encoder-decoder architecture. An encoder processes an input sequence to capture its representation, while a decoder generates an output sequence based on the



(a) Scaled Dot-Product Attention (b) Multi-head attention

Figure 4.5: Attention block diagrams as presented in [Vaswani et al. 2017].

encoded representation. A detailed description of the architecture can be found in the reference. In the literature, encoder-only or encoder-decoder architectures are used for several attention-based learning purposes.

The attention mechanism and parallel processing capability of transformers have revolutionized various NLP tasks, such as machine translation, text generation, and question-answering. Transformer-based approaches such as BERT (Bidirectional Encoder Representations from Transformers) [Devlin et al. 2018], GPT (Generative Pre-trained Transformer) [Radford et al. 2018], and T5 (Text-to-Text Transfer Transformer) [Roberts et al. 2019] have achieved state-of-the-art performance across a wide range of NLP tasks. It should be noted that transformers are not limited to NLP applications.

Having presented different neural network types, the choice of the network to be used depends on the given task, the length of the input sequence, and the available computational capabilities.

Next, we move to the use of these networks in the literature for observing the state of the vehicle. Later in this chapter, CNNs will be used in a learned observer architecture. The presented networks are the basis of the remaining work done in this thesis as they will be used as well in the next chapter for real vehicle observing applications, and in the last chapter in planning vehicle trajectories.

4.3 Learned Observers

Learning-based observers have been proposed in the literature to estimate the state of the vehicle based on measurements data. Learned observers in the literature are split into two types: hybrid approaches that combine neural networks

with either vehicle equations or model-based observers equations; and fully learned approaches that rely only on neural networks.

Hybrid approaches are as well split into two types. On the one hand, several approaches combine neural networks with a model-based observer, such as the KalmanNet [Revach et al. 2021] and its variations (e.g. [Choi et al. 2023]). The KalmanNet approach is based on a Kalman filter while replacing the Kalman gain calculation by a prediction effected by recurrent neural networks. This type of architecture was used for instance to estimate the longitudinal and lateral velocities and yaw rate of the vehicle in [Escoriza et al. 2021]. Another application to this type of hybrid observer can be seen in [Song and Fang 2021] where an architecture that combines neural networks and a sliding mode observer is proposed aiming to estimate the vehicle's velocities and yaw rate as well. Also, the work in [Liu and Guo 2021] combines the use of LSTMs with an invariant Kalman Filter to predict the position of the vehicle during GPS outages. Moreover, an encoder-only attention-based approach combined with an EKF is used in [Zhang et al. 2023] to estimate the mass of the vehicle. On the other hand, several hybrid approaches rely on combining neural networks with vehicle model equations to deliver the required estimations. For example, the work in [Graber et al. 2019] makes use of a dynamic bicycle model combined with GRU networks to estimate the side-slip angle of the vehicle.

Fully learned approaches make use of neural networks only with inputs that differ between applications to effect the required estimations. For example, the works in [Zhang et al. 2021] and [Srinivasan et al. 2020] use LSTM or GRU networks to estimate the velocities of the vehicle with different considerations to the input measurements.

Although some of the mentioned works use sensors that are not common in vehicles (e.g. throttle position sensors [Liu and Guo 2021] or longitudinal velocity measurements [Zhang et al. 2021]), others make use of standard vehicle sensors and present accurate estimations in the test scenarios they consider; the architectures proposed later-on will be compared to these.

In the remaining parts of this chapter, a CNN-based filter will be introduced and its performance will be compared to a model-based observer, before moving to real vehicle applications in the next chapter.

4.4 CNN-based observer

In this preliminary work, we aim to introduce a CNN-based vehicle position and heading observer and compare it to the Extended Kalman Filter as a way to motivate the use of learning-based observers and to highlight their advantages over model-based observers. This work is the first contribution of this thesis.

The problem that this observer aims to solve is the estimation of the state of

a kinematic bicycle model with the presence of process and measurement noise. A complete framework that includes the data generation methods and the neural network architecture is presented. In what follows, the considered system and noise are presented in Section 4.4.1, the data generation algorithms are presented in Section 4.4.2, the network architecture is presented in Section 4.4.3 and the results, including the comparison to the Extended Kalman Filter are presented in Section 4.4.4.

4.4.1 Considered system

The kinematic bicycle model, presented in chapter 2 is at the base of the proposed observer which aims to estimate its states. It will be used to generate the training data required to train the learned observer, the testing data to test the developed observer and to build the EKF used for comparison purposes later. The system is considered to be fully observable. As we seek to observe the state of the kinematic bicycle model while process and measurement noise are present, the used simulation model relies on the following discretized system:

$$z_{k+1} = z_k + f(z_k, u_k).\Delta t + w_k \tag{4.8a}$$

$$m_k = z_k + v_k \tag{4.8b}$$

where k is the time step, Δt is the time interval, m_k is the measurement, w_k is the process noise and v_k is the measurement noise. The function $f(z_k, u_k)$ refers to Equations (2.24) of the kinematic bicycle introduced previously, with z_k being the state of the kinematic bicycle model $z_k = \begin{bmatrix} X_k & Y_k & \psi_k \end{bmatrix}$ and u_k being the applied controls $u_k = \begin{bmatrix} V_k & \delta_k \end{bmatrix}$.

The introduced noise is white Gaussian with variance $\sigma'^2 = (\sigma'^2_x, \sigma'^2_y, \sigma'^2_\psi)$ for w_k and $\sigma^2 = (\sigma^2_x, \sigma^2_y, \sigma^2_\psi)$ for v_k. To assess the performance of the observer for multiple levels of noise, we will consider a varying measurement noise: the 3 standard deviations $(\sigma_x, \sigma_y, \sigma_\psi)$ are all multiplied by a scaling factor α while the process noise is constant. This implies a constant process noise covariance in the EKF with a varying measurement noise covariance. The inclusion of process noise aims to create a difference between the model used inside the EKF (kinematic bicycle model) and the simulated model (kinematic model in addition to process noise). It should be noted that the developed observer later on will be trained with $\alpha = 1$ but tested on varying α values. The considered process and measurement noise are presented in Table 4.1, the values being motivated by low cost GNSS/INS sensor properties in [Elkaim et al. 2008].

Remark 12. A kinematic bicycle simulator with a kinematic bicycle observer is chosen in this work to show the ability of learned observers to surpass the performance of model-based ones even when they are expected to well perform.

Process	
$3\sigma'_x$	0.2 m
$3\sigma'_y$	0.2 m
$3\sigma'_\psi$	3.4 mrad
Measurement	
$3\sigma_x$	1 m
$3\sigma_y$	1 m
$3\sigma_\psi$	17.4 mrad

Table 4.1: Noise parameters values. The measurement noise corresponds to $\alpha = 1$.

4.4.2 Data generation

As mentioned in the previous parts of this chapter, the training of a neural network requires the presence of training data samples. The testing will require testing data as well. For this purpose, we use the considered system above to generate data for our learned observer. We will focus on generating a well-distributed training dataset to train the learned observer, a validation dataset to assess the performance of the observer while training and a testing dataset to test the learned observer after training it. The three datasets are generated using different algorithms to confirm the performance of the developed observer. Each of the data generation procedures are detailed next.

4.4.2.1 Training data generation

We aim to create a training dataset depicting most of the behaviors of the vehicle represented by the kinematic bicycle model. To do so, we aim to have a fair distribution of the generated accelerations on the friction circle. The friction circle represents an envelope for the possible accelerations of the vehicle; and the position of an acceleration set inside the circle determines the harshness of the maneuver. We limit the considered friction circle to a maximum acceleration $|a| < 0.5g$ where the kinematic bicycle model is a valid representation of the vehicle's behavior [Polack et al. 2017].

To create the described dataset, assuming the discrete system above with a time step $\Delta T = 0.02\,\mathrm{s}$, the controls at each step should be chosen in a way that maximizes the acceleration distribution on the considered friction circle. The algorithm considers a two-dimensional array representing the friction circle. At each time step, controls between $V = 1\,\mathrm{m\,s^{-1}}$ and $V = 20\,\mathrm{m\,s^{-1}}$ for the velocity and between $\delta = -0.5\,\mathrm{rad}$ and $\delta = 0.5\,\mathrm{rad}$ for the steering angle are chosen such that the less dense parts of the friction circle are filled to result in a fair

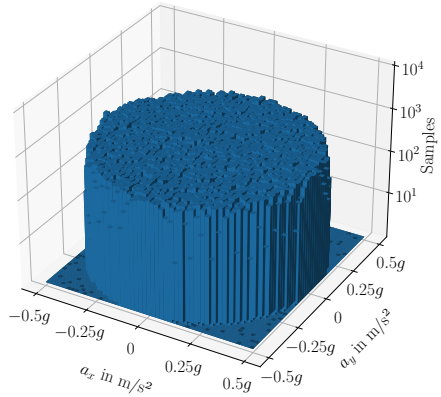


Figure 4.6: Friction circle filled using the training data generation algorithm (number of samples shown on a logarithmic axis).

distribution over the array. One thousand 40-second trajectories are generated, each starting with a vehicle state having $X = 0$, $Y = 0$ and an initial heading in the interval $\psi \in [-\pi, \pi[$. Running the algorithm generates a training dataset made of 2 million samples with a distribution on the friction circle shown in Figure 4.6. The figure shows a fair distribution of the samples over the friction circle. Measurement noise for the training data has a noise scaling factor $\alpha = 1$.

4.4.2.2 Validation and testing data generation

As mentioned previously, a validation dataset assesses the performance of the learned observer while being trained, avoiding overfitting the weights of the observer on the training set; whether a testing dataset assesses the performance of the learned observer after its training. Both datasets should be different than the training set to ensure the generalization of the performance of the learned observer on unseen scenarios.

Both sets are created using clothoid functions, resulting in C2 continuous smooth paths. After generating the paths, a pure-pursuit lateral controller is used to follow it. The pure pursuit controller was introduced in [Coulter 1992] and aims to control the steering angle of the vehicle to follow a predefined reference path. It makes use of a look-ahead distance towards the reference trajectory and relies on the angle between the vehicle's axis and the vector joining the center of the rear axle of the vehicle and the look-ahead point on the reference path. The sole use of this controller is to create the needed trajectories using the kinematic bicycle model used in this work.

For the validation data, a sinusoidal shape is created using clothoids, the path

alternates between $Y = -5\,\mathrm{m}$ and $Y = 5\,\mathrm{m}$ for an X interval of $200\,\mathrm{m}$. The resulting shape is shown in Figure 4.7; the present glitches are due to the process noise added to the model. The validation set is made of 995 data points with an added measurement noise that has a measurement scaling factor of $\alpha = 1$.

For the testing data, fifteen paths are created. These paths represent challenging maneuvers for the learned observer to be tested on. They are shown in Figure 4.8. The testing set is made of 9829 data points. Variations of the measurement noise are added to the testing set: the noise scaling factor varies between 0 and 6 with increments of 0.25 which brings the testing dataset to a total of 245,725 points.

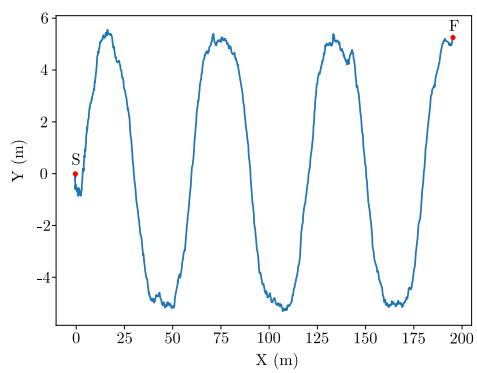


Figure 4.7: Validation trajectory.

After generating the needed datasets, the observer architecture is defined next.

4.4.3 Observer architecture

The proposed learning-based observer takes as inputs the state measurements of the position and heading of the kinematic bicycle model, in addition to the applied controls for a defined number of previous time steps; the inputs to the observer are then X_m^k, Y_m^k, ψ_m^k, V^k and δ^k for $k = t - n$, $n = 0..N$, m denoting the measurement, t being the current time, n being the considered time step, and N being total number of time steps to consider. The output of the observer is the actual state of the kinematic bicycle model at $k = t$: the simulator data without the added measurement noise.

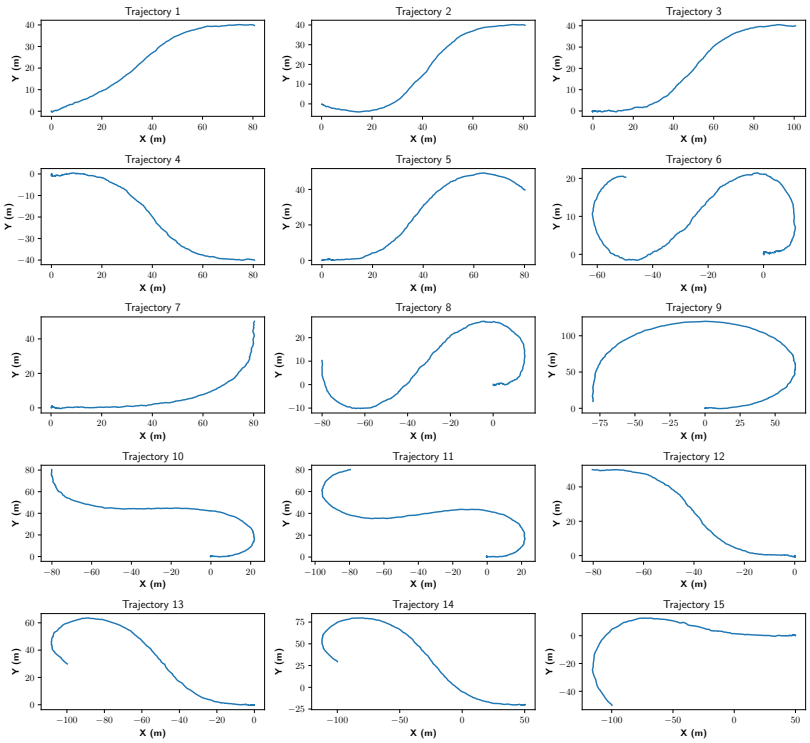


Figure 4.8: Testing trajectories.

We propose a CNN-based observer which will be compared to an LSTM-based observer and to an EKF based on specific metrics defined later-on. For each of the learned observers (CNN and LSTM) we will consider input window sizes of $N = 20$, $N = 40$, $N = 60$ and $N = 80$ time steps: a total of 8 learned observers. The motivation behind each type of neural network, in addition to the details of the architecture used are detailed next.

4.4.3.1 CNN-based observer

As presented in Section 4.2.4, CNNs are neural networks whose architecture involves a series of convolution and pooling layers, processing data with a grid-like topology, as the measurements and inputs for different time steps considered in this work. The weight sharing property of CNNs motivates their use in the learned observer we introduce. In our context, weight sharing is used as a way to similarly extract important properties from the different time steps of each variable (either measurements or controls) in the first layers of the network, to treat them separately, then jointly in the following layers. Thus, the used architecture will involve two 5x1 convolutions first, followed by a max pooling layer of 4x1, then the

features are treated jointly by two consecutive 1x3 convolutions. Two fully connected (feedforward) layers connect the last CNN feature map to the observer's output. The described architecture is seen in Figure 4.9.

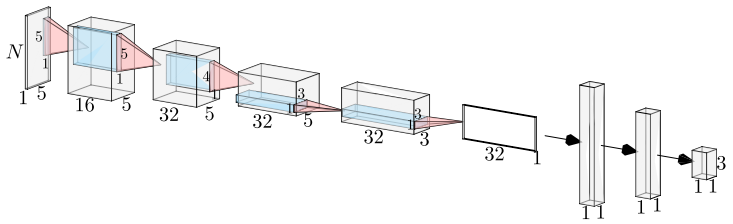


Figure 4.9: Proposed CNN architecture.

4.4.3.2 LSTM-based observer

As presented in Section 4.2.5.2, LSTMs treat sequences of data and are able to recognize temporal dependencies and patterns in time series, which makes them a good candidate for observing applications. This motivates their comparison to the CNN-based approach and to the EKF later on. The proposed architecture involves four consecutive LSTM layers with 8, 16, 32, and 32 neurons respectively, followed by 2 fully connected (feedforward) layers. The described architecture is seen in Figure 4.10.

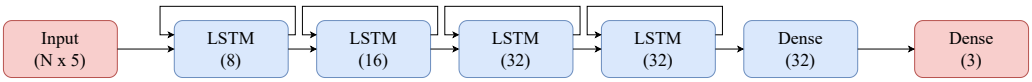


Figure 4.10: Tested LSTM architecture.

For both architectures, the loss function used is the mean squared error, comparing the output of each observer to the kinematic bicycle simulator data before measurement noise addition. The weights of the network are initialized using the Xavier initialization technique introduced earlier. Hyperparameter optimization is done using grid search for $N = 20$ and then applied to the remaining observers.

The presented learned observers are implemented and trained using Keras[1] on an Nvidia Geforce GTX 1650 Ti. Early stopping is used to avoid overfitting. The performance of the learned observers is analyzed next.

4.4.4 Results and analysis

The trained observers are tested on the generated testing dataset presented before. It should be noted that the testing dataset is never seen by the observers while

[1] https://keras.io/

training and the levels of noise present in it are higher than the ones the observers were trained on. The testing procedure will start by comparing the performance of each type (CNN or LSTM) of the observers for the different window sizes $(20, 40, 60, 80)$ while varying the noise scaling factor α; then, the best-performing observers of each type are compared with the EKF to conclude the study. The metric used for comparison is introduced next, followed by the comparisons between different observers.

4.4.4.1 Metric

The comparison of the performance of the observers involves the comparison of their estimations to the ground truth data. The presented learned observers aim to estimate the state of the kinematic bicycle model, making the error vector three-dimensional. The proposed metric reduces the error vector to a scalar allowing to describe the performance of the observer using a single value. As the goal of this work is to motivate the use of learned observers and show their advantages over model-based observers, the considered metric takes as reference the errors of the EKF at $\alpha = 1$. In addition, the introduced metric gives similar weight to the three variables. Then, the metric introduced is a normalized root mean square error (NRMSE) defined as:

$$\text{NRMSE} = \sqrt{w_x E_x^2 + w_y E_y^2 + w_\psi E_\psi^2} \tag{4.9}$$

E_x^2, E_y^2, E_ψ^2 being the mean squared error for each of the three variables and w_x, w_y, w_ψ being the weights given to each variable based on the reference case of the EKF, such that:

$$w_x = \frac{1}{3E_{\text{ref},x}^2} \tag{4.10a}$$

$$w_y = \frac{1}{3E_{\text{ref},y}^2} \tag{4.10b}$$

$$w_\psi = \frac{1}{3E_{\text{ref},\psi}^2} \tag{4.10c}$$

with $E_{\text{ref},x} = 0.24\,\text{m}$, $E_{\text{ref},y} = 0.23\,\text{m}$ and $E_{\text{ref},\psi} = 4.1\,\text{mrad}$ being the errors of the EKF for the low noise testing dataset.

The defined metric is used to compare the different observers next.

4.4.4.2 CNN-based observers comparison

The defined metric is applied to the predictions of the four CNN observers on the testing dataset. The plot in Figure 4.11 presents the performance of the learned observers with respect to noise increase. It can be seen that for all observers, the

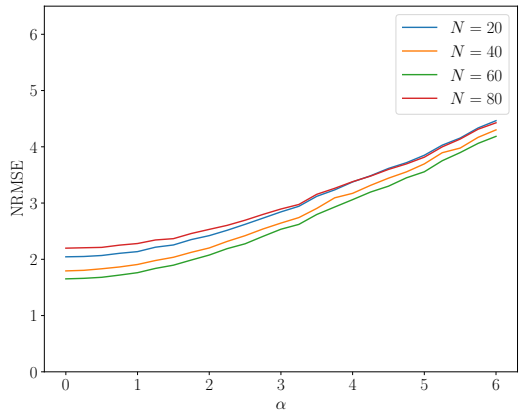


Figure 4.11: CNN-based observers performance with respect to noise. The $N = 60$ observer shows the lowest errors.

performance deteriorates with the increase of noise level. Similar performance can be seen for the $N = 20$ and $N = 80$ observers while the $N = 60$ observer is able to deliver the lowest errors for the whole noise range. The $N = 60$ CNN observer will then be used for the comparison with the EKF.

4.4.4.3 LSTM-based observers comparison

The defined metric is also applied to the predictions of the four LSTM observers on the testing dataset. The plot in Figure 4.12 presents the evolution of the performance of the observers with respect to noise increase. The performance of the $N = 20$ and $N = 80$ observers is close while the $N = 60$ observer shows the highest errors. The lowest errors are delivered by the $N = 80$ observer that will be used in the comparison with the EKF.

4.4.4.4 Comparison with the EKF

After comparing the different learned observers of each category, the best performing ones are compared with the EKF. Tables 4.2 and 4.3 show the root mean squared errors (RMSE) of the three predicted variables by each of the observers for $\alpha = 1$ and $\alpha = 6$ respectively. The tables show us that the EKF delivers the lowest errors for the low noise case while both learned observers are able to outperform it for the high noise case. It can be seen also that the CNN-based observer delivers lower X and Y errors than the LSTM-based observer while the ψ performance is close.

The defined NRMSE metric is used to compare the CNN, LSTM and EKF observers. The performance of the three observers is plotted in Figure 4.13 while varying the noise scaling factor α. The plot shows that for lower noise, the EKF is

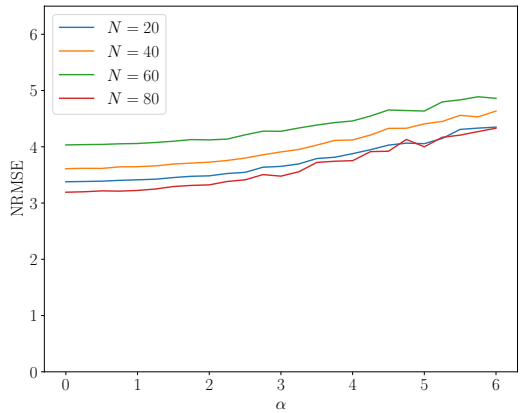


Figure 4.12: LSTM-based observers performance with respect to noise. The $N = 80$ observer shows the lowest errors.

Variable	**EKF**	**CNN**	**LSTM**
X (m)	0.24	0.28	0.65
Y (m)	0.23	0.23	0.90
ψ (mrad)	4.1	11	13

Table 4.2: RMSE comparison of the observers on low noise data ($\alpha = 1$) showing that the EKF observer outperforms both learned observers.

Variable	**EKF**	**CNN**	**LSTM**
X (m)	1.56	0.91	0.98
Y (m)	1.88	0.89	1.10
ψ (mrad)	26	20	20

Table 4.3: RMSE comparison of the observers on high noise data ($\alpha = 6$) showing that the CNN observer outperforms both the EKF and LSTM observers.

able to have lower errors than both learned observers while its performance deteriorates almost linearly with higher noise scaling factors, showing lower robustness to noise. Both learned observers perform best at lower noise, which is logical as they have been trained on low noise data. Both will surpass the performance of the EKF with noise increase, the CNN observer at $\alpha = 2$ and the LSTM observer at $\alpha = 3$. The CNN observer has lower NRMSE scores than the LSTM for all of the error levels.

A visualization of the performance difference between the EKF and the CNN

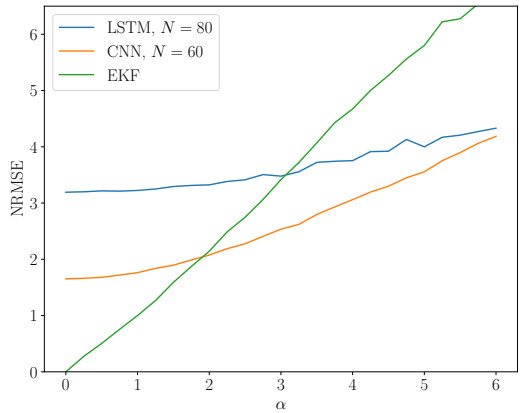


Figure 4.13: Comparison between the performance evolution of the learning-based observers and the EKF with respect to noise. The EKF shows the lowest errors at low noise levels, but it is outperformed by both learned observers with noise increase.

for $\alpha = 3$ is shown in Figure 4.14: the CNN estimations are closer to the real trajectory than that of the EKF.

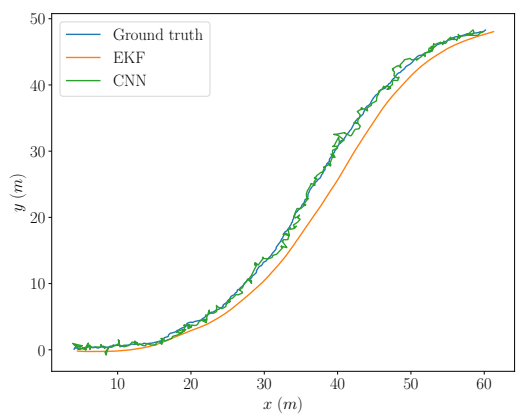


Figure 4.14: Visualization of the estimations of the CNN and the EKF. Although the EKF is smoother, the CNN is more accurate.

4.4.4.5 Discussion

In brief, the proposed CNN architecture is able to outperform the LSTM one for all the levels of noise, while it outperforms the EKF for a specified noise domain. Model-based observers are able to adapt when the measurements are reliable (low measurement noise) while both the presented learned observers deal better with

higher levels of noise.

4.5 Conclusion

In this chapter, we introduced several neural networks types while discussing the functioning of each of them. The use of these networks in the literature for state observation purposes was then discussed while defining the aim and methodology of each application.

As this chapter is considered an introduction to the use of neural networks in vehicle state observers, a simple observing application was applied to the kinematic bicycle model motivating the application of learned observers. The proposed learned observer was compared to a model-based state-of-the-art observer, the EKF. The comparison showed more robustness of the proposed approach to noise in contrast with the EKF. The EKF seemed able to provide accurate estimations in low measurement noise conditions, while both learned observers were able to easily outperform it with noise increase.

The advantages of learned observing strategies over model-based ones in this chapter focused on providing reliable estimations through high-noise situations. In the next chapter, these advantages will include as well overcoming the difficulties related to model validity and model parameter identification. This will be shown with an application to real vehicles.

Chapter 5

Learned Observers Applied to Real Vehicles

Contents

5.1 Introduction **82**

5.2 Stadtpilot vehicle description **82**

5.2.1 Overview 82

5.2.2 Sensor setup 82

5.3 Data collection **85**

5.4 Vehicle velocity and yaw rate observer **86**

5.4.1 Proposed architecture 88

5.4.2 Results 90

5.5 Vehicle side-slip angle observer **100**

5.5.1 Proposed architecture 100

5.5.2 Results 101

5.6 Conclusion **107**

5.1 Introduction

In the previous chapters, vehicle models used to describe the behavior of the vehicle were presented and then were used in model-based observers for vehicle state estimation. Knowing the problems associated with model-based observers, learning-based observers were introduced as a method to reach accurate state observations.

So far, the presented explanations and difficulties in the model-based observers chapter were not explicitly demonstrated, while the developed work in the learning-based observers chapter was simulator based. This chapter will implement the observing concepts presented in the previous chapters to real vehicles. Additional learned observing techniques will be introduced. The aim is to show the limitations of model-based observers and the advantages of learning-based ones in real-life scenarios, especially in high dynamics.

In this chapter, the used test vehicle will be described and used for data collection in well-defined scenarios. The collected data will then be used to implement two applications of learned observers: a velocity observer and a side-slip angle observer, considered as additional contributions by this thesis. The proposed observers will be compared to state-of-the-art model-based and learning-based observers.

The tests presented in this chapter were conducted in the labs of the Institute of Control Engineering in the Technical University of Braunschweig[1], Germany.

5.2 Stadtpilot vehicle description

5.2.1 Overview

As we aim to prove our claims on model-based and learning-based observers in real scenarios, we perform our tests on the Stadtpilot[2] vehicle. The AUDI A6 Avant C7 vehicle shown in Figure 5.1 is part of the project aiming to effect fully autonomous maneuvers in the urban areas of the city of Braunschweig. The characteristics of the vehicle, as communicated by the control engineering institute in Braunschweig are shown in Table 5.1. The sensor setup in the vehicle is presented next.

5.2.2 Sensor setup

As stated before, access to ground truth data in real vehicles is not as simple as it is in simulated environments and is usually assumed to be granted through high-precision sensors. The considered vehicle is equipped with a dual-antenna deeply-

[1] https://www.tu-braunschweig.de/ifr
[2] https://www.tu-braunschweig.de/stadtpilot

Figure 5.1: The Stadtpilot vehicle.

Parameter	Description	Value
M	Mass of the vehicle	1578 kg
l_{f}	length from CoG to the front axle	1.134 m
l_{r}	length from CoG to the rear axle	1.578 m
b	Track width	1.513 m
I_z	Moment of inertia around the z-axis	2924 kg m^2

Table 5.1: Parameters of the vehicle used for data collection (CoG: center of gravity).

coupled INS/GNSS iTraceRT F400[3] sensor shown in Figure 5.2. The sensor is able to measure the position (Easting – X , Northing – Y) in UTM(Universal Transverse Mercator)-coordinates, the longitudinal and lateral velocities (V_x, V_y), the longitudinal and lateral accelerations (a_x, a_y), the yaw (ψ), pitch (θ), and roll (Φ) angles and rates ($\dot{\psi}$, $\dot{\theta}$, $\dot{\Phi}$), and the side-slip angle (β) with the accuracy shown in Table 5.2 based on the sensor's technical sheet. This sensor is considered the ground truth for our system.

In addition to the installed high precision sensor, the vehicle is equipped by default with the Audi Sensor Array (SARA) which consists of an IMU providing measurements of the accelerations (a_x, a_y) and the yaw rate ($\dot{\psi}$) in addition to wheel speeds (W_{ij}) and steering angle (δ) sensors. These would be referred to as the standard in-car sensors which are expected to deliver lower accuracy measurements. Neither the Institute in Braunschweig could present any information

[3] https://www.imar-navigation.de/en/products/by-product-names/item/itracert-f200-itracert-f400-itracert-mvt

Figure 5.2: The iTraceRT F400 sensor.

Measure	**Accuracy**
Position	$\pm 2\,\mathrm{cm}$
Velocity	$\pm 0.02\,\mathrm{m\,s^{-1}}$
Acceleration	$\pm 1\,\mathrm{mg}$
Roll/Pitch/Heading	$\pm 0.01°$
Side-slip	$\pm 0.1°$

Table 5.2: Accuracy of the measurements provided by the iTraceRT sensor.

about the accuracy of these sensors, nor any information could be found online. In comparison with the iTraceRT measurements, the errors of the longitudinal and lateral accelerations and yaw rate of the SARA are presented in Table 5.3 with the low dynamics column referring to the combination of maneuvers effected in the city of Braunschweig with lateral accelerations $a_y^{\max} < 0.5\,\mathrm{g}$ and the high dynamics column referring to the combination of maneuvers effected on the test track with lateral accelerations $a_y^{\max} > 0.5\,\mathrm{g}$: the two types of maneuvers are detailed in the next section.

The sensor setup in the test vehicle is shown in Figure 5.3 while visualizing the variables of interest in our observing applications: the longitudinal and lateral

Measures	Accuracy	
	Low Dynamics	High Dynamics
	Mean / Std.	Mean / Std.
$a_x(\mathrm{m\,s^{-2}})$	0.12 / 0.08	0.26 / 0.22
$a_y(\mathrm{m\,s^{-2}})$	0.25 / 0.19	0.81 / 0.7
$\dot{\psi}(\mathrm{mrad})$	2.41 / 3.54	15.4 / 17.3

Table 5.3: SARA accuracy in comparison with the iTraceRT sensor.

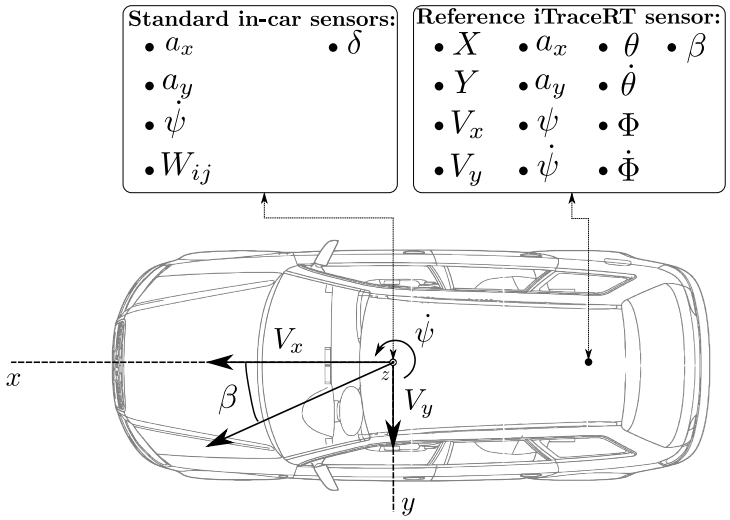


Figure 5.3: Top view of the test vehicle showing the available in-car and reference sensors.

velocities, the yaw rate and the side-slip angle at the center of gravity of the vehicle. The standard in-car sensors provide measurements at a 50 Hz frequency while the iTraceRT sensor provides measurements at 100 Hz. In our application, the measurements provided by the iTraceRT sensor are down-sampled to 50 Hz for synchronization purposes.

The aim of the developed observers in this chapter would be to rely on standard in-car sensors to be able to provide estimations as close as possible to the reference ground truth sensor.

Having defined the sensor setup of the vehicle, the data collection process using the defined sensors is presented next.

5.3 Data collection

As we aim to develop learned observers, data is needed for training the proposed architectures and for testing them later on. The system defined in the previous section is used for data collection. We seek to collect a well-distributed dataset reflecting the wide range of possible behaviors of the vehicle. For this purpose, the data collection procedure is divided into two main types: low acceleration maneuvers depicted by inner-city driving in Braunschweig, Germany and high acceleration maneuvers depicted by driving on a dedicated test track near Peine, Germany. The test track is shown in Figure 5.4 and is basically an old airport runway that is currently used for autonomous vehicle testing by several German institutes.

Figure 5.4: Test track used for effecting high acceleration maneuvers.

The inner-city driving included multiple maneuvers as U-turns and lane changes, in addition to normal driving throughout the city resulting in 770,000 samples equivalent to 4.3 hours of driving. While the harsh maneuvers effected on the test track resulted in 300,000 samples equivalent to 1.6 hours of driving. The total dataset consists of about 1 million data samples.

To analyze the characteristics of the collected dataset, its samples are plotted on the friction circle in Figure 5.5. The plot shows that most of the effected maneuvers are within the low acceleration range which corresponds to normal city driving and other maneuvers at low speed while other samples are in the higher acceleration range (reaching $a = 1g$), those correspond to the harsh maneuvers effected on the test track. The distribution is biased towards lower accelerations due to the ease of collecting data at lower accelerations and the difficulty of collecting data at the limits of handling; this is also reflected in the distribution of the side-slip angles over the collected dataset shown in Figure 5.6. It can be seen that the dataset contains a near-symmetrical distribution of side-slip angles in a range of $\pm 18°$. Given the presented data properties, we will analyze in our approaches the performance of the observers in high dynamic scenarios specifically.

The collected dataset is subject to a 60%-20%-20% train-validation-test split. The split is carefully performed to include both low acceleration and high acceleration data in all sets.

After collecting the needed datasets to train and evaluate our approaches, the different observers are presented next.

5.4 Vehicle velocity and yaw rate observer

As mentioned previously, one of the targets of our developed observers is to estimate the longitudinal and lateral velocities and yaw rate of the vehicle. These quantities are key to describing the behavior of the vehicle as seen in chapter 2

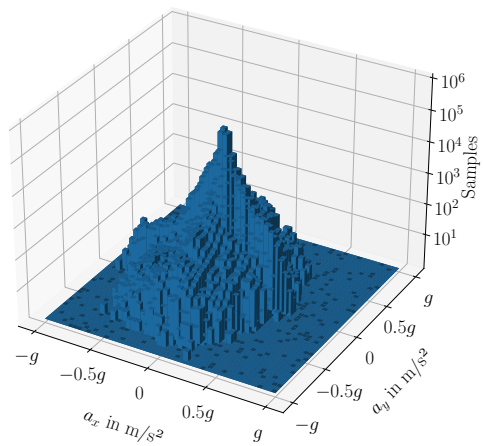


Figure 5.5: Distribution of the collected data on the friction circle (number of samples shown on a logarithmic axis): samples are present in both low acceleration and high acceleration ranges with a bias towards lower accelerations. g is the gravitational acceleration expressed in $\mathrm{m\,s^{-2}}$.

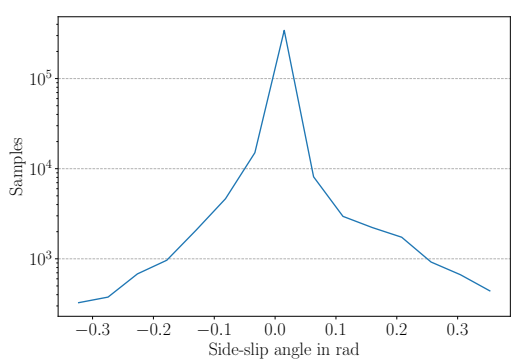


Figure 5.6: Distribution of the side-slip angles over the collected dataset (number of samples shown on a logarithmic axis). Small side-slip angle data is easier to collect which explains the distribution being biased towards smaller angles.

and they were a subject of interest of many researchers in the literature as seen in chapters 3 and 4. The longitudinal and lateral velocities are not easily accessible within vehicles while the yaw rate is usually measured but subject to noise. The aim is to estimate accurately the three variables.

To estimate the mentioned quantities, an LSTM-based approach is considered and is presented in Section 5.4.1; after training the proposed network, it is tested in Section 5.4.2. To be able to assess the performance of the proposed observer in detail, it will be compared to state-of-the-art model-based and learning-based observers presented in chapters 3 and 4 for low dynamic and high dynamic maneuvers in Section 5.4.2.

Remark 13. As opposed to the previous application, a CNN-based approach was not able to converge to accurate results when training. This may question the usability of CNN-based observers outside filtering applications. Further experiments would be needed to reach a conclusion.

5.4.1 Proposed architecture

As stated previously, the goal of the proposed observer is to estimate the longitudinal and lateral velocities and yaw rate of the vehicle using the standard in-car sensors only. Having the iTraceRT measurements considered as ground truth, the network could be trained and its output values could be compared to the reference sensor values.

As recurrent neural networks are able to capture temporal dependencies between different time steps, they constitute a good candidate for our observing application. As presented in chapter 4, LSTMs were introduced as a variant of recurrent neural networks to solve the vanishing/exploding gradient problem. LSTMs are implemented in our observer architecture.

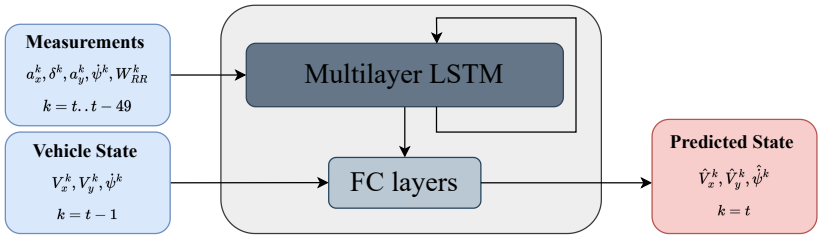
The proposed observer architecture takes two inputs:

- The first input consists of the in-car sensor measurements for the 50 previous time steps. The length of the used sequence was decided using the same methodology employed in the CNN work presented in chapter 4. A single rear-wheel measurement was chosen from the in-car sensors to ensure a fair comparison with state-of-the-art methods that do the same.

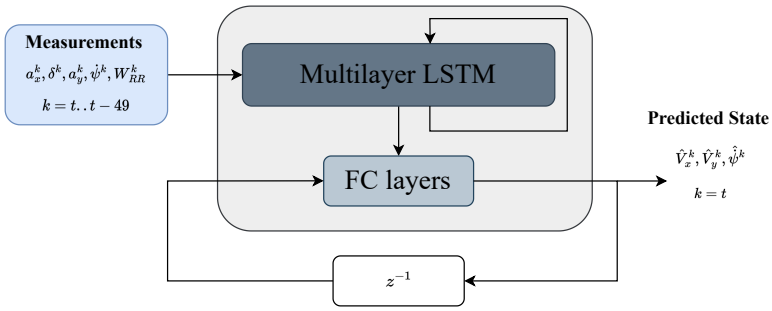
- The second input consists of the previous vehicle velocities and yaw rate providing the observer knowledge about the state in the previous time step. The source of this input differs between training and testing as it is explained below.

The output of the observer are the current estimations of the velocities and yaw rate of the vehicle.

Two operating modes of the proposed architecture are specified:



(a) Training mode: the network takes as inputs the standard in-car sensor measurements and the vehicle velocities and yaw rate at the previous time step.



(b) Testing mode: the network takes as inputs the standard in-car sensor measurements and the predictions of the observer at the previous time step. (z^{-1} represents a time delay of one step).

Figure 5.7: The two operating modes of the proposed architecture. (FC: fully connected)

- The training mode shown in Figure 5.7a: in this mode, the vehicle state from the previous time step (second input) fed to the network is the ground truth data to which Gaussian noise is added. The purpose of adding noise to the ground truth data is to make the network immune to drift as it will adapt to predict an accurate state estimation even if errors are present in the second input which could be the case in the testing mode. The standard deviations of the added noise are $0.03\,\mathrm{m\,s^{-1}}$ for both longitudinal and lateral velocities and $0.003\,\mathrm{rad\,s^{-1}}$ for the yaw rate.

- The testing mode shown in Figure 5.7b: in this mode, the second input representing the velocities and yaw rate of the vehicle at the previous time step is fed from the previous observer prediction, which creates a closed loop as seen in the Figure. At the start of the observing operation, an initial value should be given to the observer.

The multilayer LSTM block is made of four LSTM layers including 32, 64, 64 and 128 neurons respectively. Fully connected layers follow to which the outputs

of the last LSTM layer and the previous states are fed. The three fully connected layers consist of 64, 128 and 64 neurons respectively.

All the inputs and outputs of the network are scaled between 0 and 1. The same scaler is used for the output values and the previous state input values. The fully connected layers have sigmoid activation functions. Weights are initialized using Xavier initialization. An L2 loss function is used when training to compare the outputs of the network to the values of the iTraceRT sensor.

The network is implemented using PyTorch[4] and is trained for 50 epochs (after early stopping) using Adam[5] on an Nvidia Geforce GTX 1650 Ti.

After training the network, tests are performed in the next section to evaluate its performance. The trained network is able to deliver estimations at 150 Hz.

5.4.2 Results

The performance of the proposed observer is tested in this Section. To be able to assess the accuracy of the approach, it will be compared to several state-of-the-art model-based and learning-based observers presented in the previous chapters. The choice of the state-of-the-art approaches to be compared to the proposed method is explained in Section 5.4.2.1. The result analysis is then split into 3 parts: the overall performance depicting the errors of the observers for the whole testing set in Section 5.4.2.2, and the low and high dynamic maneuvers depicting the performance of each of the observers for specific maneuvers in Sections 5.4.2.3 and 5.4.2.4 respectively. The proposed testing procedure will assess the performance change of each of the observers in different driving situations.

5.4.2.1 State-of-the-art observers

The performance of the proposed observer is compared to model-based and learning-based observers presented in the literature. Two model-based and two learning-based observers are chosen.

The chosen model-based observers reflect two levels of vehicle modeling complexities: a dynamic bicycle model with an adaptive linear tire model observer [van Aalst et al. 2018] referred to as DBM, and a four-wheel vehicle model with a Pacejka tire model observer [Katriniok and Abel 2016] referred to as 4WM. Both observers make use of the EKF for their estimations. The decision to pick these two works is led by two main motivations: first, they showed the capability of delivering accurate estimations in the scenarios they were tested on; second, they present two models with different validity domains which will infer the analysis of the effects of using simpler models in more complex maneuvers.

[4] https://pytorch.org/
[5] https://pytorch.org/docs/stable/generated/torch.optim.Adam.html

Both model-based observers were implemented following the equations of the dynamic bicycle model and the four-wheel model provided in Chapter 2 and the equations of the EKF provided in Chapter 3. The parameters used for both models, including the tire parameters, are communicated by the Institute and they were identified formerly through a series of tests and observations. It should be noted that errors or changes in these parameters over time, especially for the tire parameters, may have great effects on the performance of the model-based observers as it is seen afterwards. This further motivates the use of learned techniques.

Remark 14. The tuning of the process and measurement noise covariances of the EKF is done as specified in each of the considered works. This applies for all the state-of-the-art implementations in this thesis.

The learning-based observers are chosen to represent a hybrid approach and a fully learned approach. The hybrid approach is the KalmanNet-based velocity observer [Escoriza et al. 2021] referred to as KN; the KalmanNet architecture defined previously has gained popularity in several observing applications. The chosen KN approach estimates the needed quantities using recurrent neural networks and calculations based on the standard in-car sensor measurements. The fully learned approach is one that uses GRU networks for velocity estimation [Srinivasan et al. 2020] referred to as GRU; it uses 200 previous measurements with no information about the previous state. Both approaches were developed to estimate the velocities in autonomous racing applications. They are expected to deliver accurate estimations even in high dynamics.

Both learning-based observers were implemented using PyTorch and were trained on the same training data that our approach is trained on.

5.4.2.2 Overall performance

The metric we use to evaluate the performance of the observers is the mean absolute error (MAE). The notion of ranking stated in the following paragraphs consists of ranking first the observer with the lowest errors and ranking last the observer with the highest errors. The evaluation of the proposed method starts by comparing its errors to the considered state-of-the-art observers for the whole testing set before moving in the next sections to specific scenarios.

For each variable the MAE of the estimations of each observer with respect to the reference sensor is calculated. The results are shown in Table 5.4.

The presented errors show that the proposed method is able to deliver the lowest errors among state-of-the-art observers for all of three variables. The KN approach ranks second and the DBM approach ranks last for the longitudinal velocity estimation; the GRU approach ranks second and the KN approach ranks last for the lateral velocity estimation; the DBM approach ranks second and the

State	DBM	4WM	KN	GRU	Ours
V_x (m/s)	0.2	0.072	0.045	0.054	**0.040**
V_y (m/s)	0.052	0.038	0.095	0.023	**0.021**
$\dot{\psi}$ (mrad/s)	4.51	4.52	4.54	4.68	**2.94**

Table 5.4: Mean absolute error for the different observers calculated for the whole testing set. The errors of the proposed approach are the lowest among the state of the art observers. **Color code:** Green cells indicate the lowest errors. Orange cells indicate the next to lowest errors. Red cells indicate the largest errors.

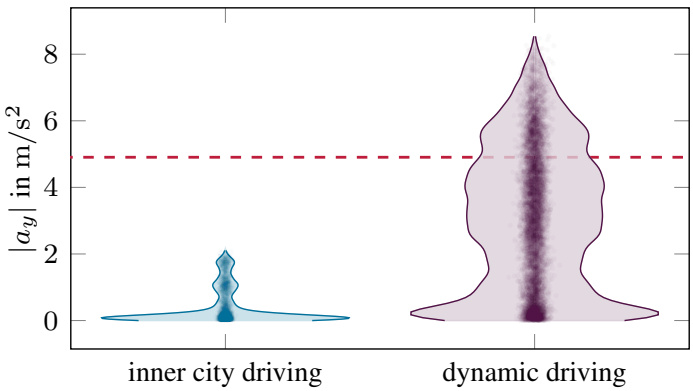


Figure 5.8: Violin plot showing the lateral accelerations distribution in the considered test sets. The dynamic driving shows significantly higher accelerations depassing the 0.5 g limit presented in [Polack et al. 2017].

GRU approach ranks last for the yaw rate estimation. To further inspect the performance of the different observers two scenarios are considered next: a low dynamic scenario and a high dynamic scenario.

5.4.2.3 Low dynamic maneuver

To inspect in detail the performance of the different observers, a non-dynamic maneuver is considered. The lateral acceleration distribution of the considered maneuver is shown in Figure 5.8 denoted as inner-city driving. The maximum lateral acceleration reached in the considered maneuver is $a_y^{\max} = 2.1\,\mathrm{m\,s^{-2}}$ indicating low dynamic driving.

The MAE for the different observers in the considered maneuvers are shown in Table 5.5. It is clear that the proposed approach is able to deliver the lowest errors for all of the three variables among the considered state-of-the-art approaches. It can be seen that for the longitudinal velocity estimation the KN approach shows

errors close to our approach, while the DBM approach shows the highest errors. For the lateral velocity estimation, the GRU approach ranks second while the KN approach ranks last with significant errors in comparison to our approach. The errors are quite similar for the yaw rate estimation except for the GRU approach that delivers the highest errors.

State	DBM	4WM	KN	GRU	Ours
V_x (m/s)	0.12	0.084	0.041	0.059	**0.039**
V_y (m/s)	0.049	0.038	0.055	0.014	**0.011**
$\dot{\psi}$ (mrad/s)	2.15	2.48	2.52	4.02	**2.11**

Table 5.5: Mean absolute error for the different observers calculated during low dynamic driving. The errors of the proposed approach are the lowest among the state of the art observers. Check Table 5.4 for color codes.

To further inspect the presented results, a plot showing the estimations of the proposed approach and the next best performing approach for each of the variables for a sequence in the highest lateral acceleration region of the considered maneuver is shown in Figure 5.9. The other approaches are discarded for visibility reasons. The plot shows the different estimations with respect to the reference sensor. The reason behind choosing a high acceleration region is that higher accelerations are associated with more challenging estimations, especially for model-based observers. The plot shows a close performance between the proposed method and the KN approach for the longitudinal velocity estimations and with the DBM approach for the yaw rate estimations, while the proposed approach delivers estimations closer to the reference for the lateral velocity estimation resulting in an advantage over the GRU approach.

As the shown plot does not present the performance of all the observers at once for visibility issues, a decreasing distribution of absolute errors is presented in Figure 5.10 showing the decreasingly sorted errors of all the observers. The plot reflects the values seen in the MAE table: the proposed approach has an advantage over the remaining approaches and is close to the next best-performing approaches for each of the estimated variables. None of the considered state-of-the-art approaches is able to achieve low errors for the three variables simultaneously while the presented approach delivers the most accurate estimations for all variables.

To inspect the behavior of the observers in a more challenging scenario, a high dynamic maneuver is considered next.

5.4.2.4 High dynamic maneuver

The performance of the observers in a high dynamic maneuver is considered in this Section. The considered dynamic maneuver involves the lateral acceleration

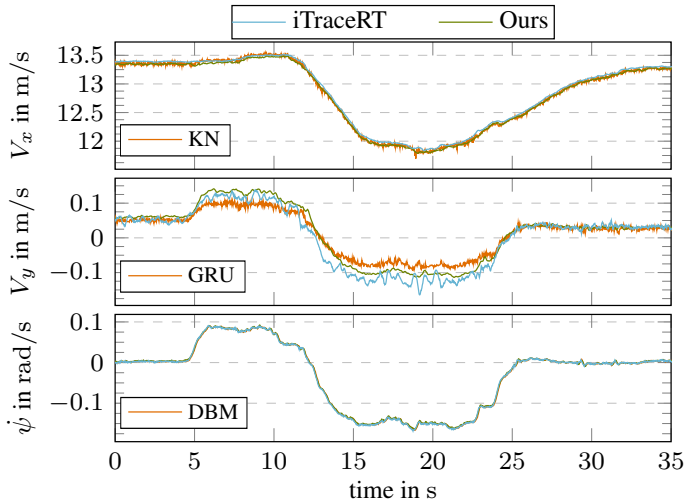


Figure 5.9: Comparison between the performance of the proposed approach, the next best performing approach and the ground truth sensor in low dynamic driving. The estimations are close for V_x and $\dot{\psi}$ but an advantage for the proposed method is present for V_y.

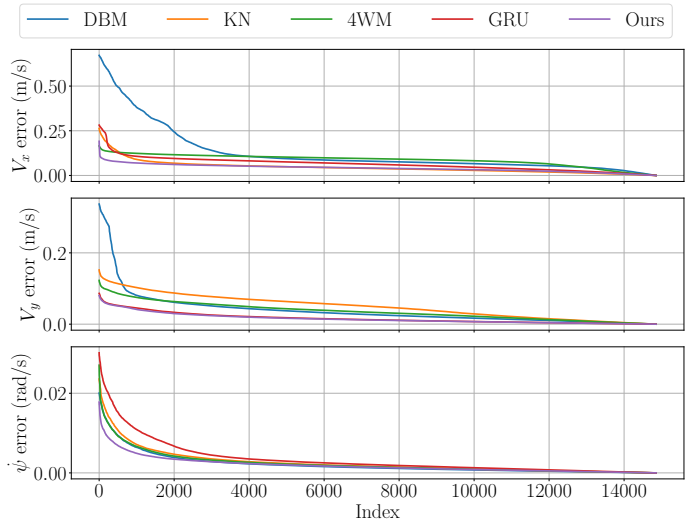


Figure 5.10: Error distribution plot for the proposed approach and the considered state-of-the-art observers for the low dynamic maneuver. The proposed approach shows an advantage over the other approaches.

distribution shown in Figure 5.8. The Figure shows accelerations reaching $a_y^{\max} =$

0.8 g exceeding the 0.5 g limit and representing harsh maneuvers. In this type of scenarios, the dynamic bicycle model with an adaptive linear tire is expected to lose validity. The other observers are expected to present lower performance as well.

The MAE for the different considered maneuvers are shown in Table 5.6. The errors associated with all the observers increase in comparison with the previous maneuver. This is due to the harshness of the maneuver. The proposed approach delivers the lowest errors among the considered state-of-the-art approaches. The GRU approach ranks second for both the longitudinal and lateral velocity estimations while the DBM approach delivers the highest errors. The DBM approach ranks second for the yaw rate estimation while the KN approach shows the highest errors.

The GRU approach ranks second in both low and high dynamic maneuvers for the lateral velocity estimation. Also, the DBM approach ranks second in both maneuvers for the yaw rate estimation. For the longitudinal velocity estimation, the KN approach ranks third in the harsh maneuver while it ranks second in the normal driving maneuver; the GRU approach ranks second in the harsh maneuver while it ranks third in the normal driving maneuver. The proposed observer ranks first for all the variables in both maneuvers.

State	DBM	4WM	KN	GRU	Ours
V_x(m/s)	0.48	0.13	0.10	0.091	**0.079**
V_y(m/s)	0.18	0.10	0.10	0.068	**0.065**
$\dot{\psi}$(mrad/s)	15.8	17.0	18.5	16.1	**9.2**

Table 5.6: Mean absolute error for the different observers calculated for high dynamic driving. The errors of the proposed approach are higher than the previous driving maneuver but are the lowest among the state-of-the-art observers. Check Table 5.4 for color codes.

The same visualization procedure employed in the previous section is used to compare the performance of the different observers. The estimations of the proposed approach and the next best-performing approach for each of the variables are shown in Figure 5.11 for a sequence in the highest lateral acceleration region of the considered maneuver. The iTraceRT reference values are shown as well. The longitudinal velocity estimations are close for the proposed approach and the GRU approach with a slight advantage to our method. The lateral velocity and yaw rate estimations show that the proposed approach is able to follow the ground truth values in a more accurate way.

It is remarkable that the DBM approach shows significantly high errors for the longitudinal and lateral velocity estimations, this is due to the use of an invalid model in the observer. The 4WM approach shows lower errors than the DBM

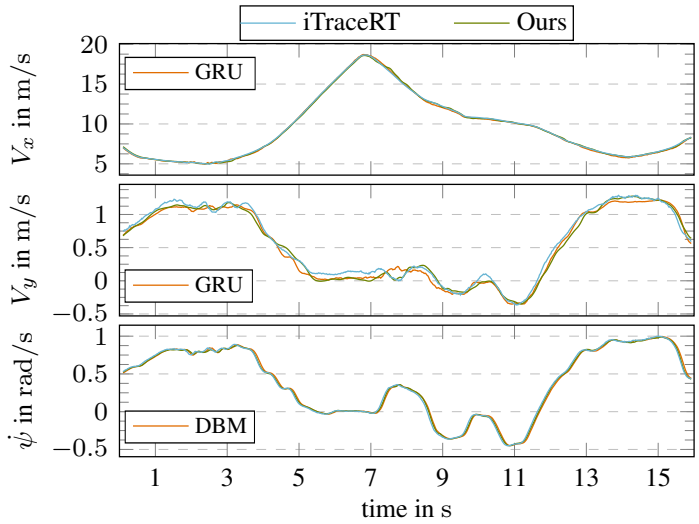


Figure 5.11: Comparison between the performance of the proposed approach, the next best performing approach and the ground truth sensor in high dynamic driving. The estimations are close for V_x but a clear advantage for the proposed method is present for V_y and $\dot{\psi}$.

approach for the longitudinal and lateral velocities but higher errors than the two fully learned approaches, this can be due to the loss of robustness in the high dynamic scenarios. Usually, the tire model parameters are not easily identified and inaccurate parameters can lead to inaccuracies in the observer's behavior.

For the same reasons mentioned in the previous section, the decreasing distribution of absolute errors is presented in Figure 5.12 showing the decreasingly sorted errors of all the observers. The presented plot shows that the presented approach is able to present the lowest errors among other approaches although some outliers are present for the longitudinal velocity estimation. None of the other approaches delivers accurate estimations for all of the three variables.

Note that the MAE of the different observers for all the trajectories of the testing set are presented in Tables 5.7-5.9.

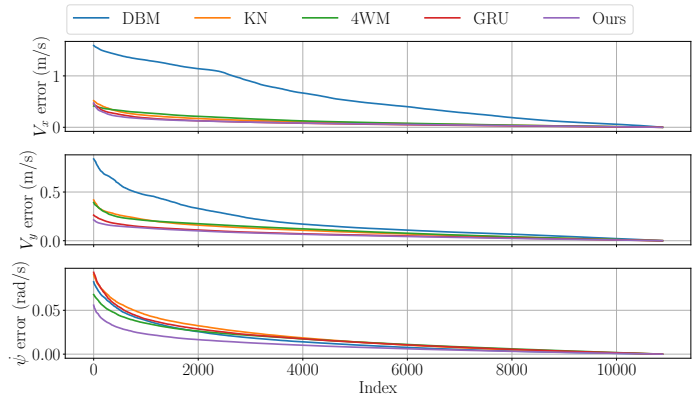


Figure 5.12: Error distribution plot for the proposed approach and the considered state-of-the-art observers for the high dynamic maneuver. The proposed approach shows an advantage over the other approaches although outliers can be seen for the longitudinal velocity estimation.

Trajectory	$a_y^{\max}$	DBM	4WM	KN	GRU	Ours
1	$1.82\,\mathrm{m\,s^{-2}}$	0.095	0.04	0.035	0.044	**0.029**
2	$2.45\,\mathrm{m\,s^{-2}}$	0.11	0.047	0.039	0.052	**0.038**
3	$2.1\,\mathrm{m\,s^{-2}}$	0.12	0.084	0.041	0.059	**0.039**
4	$3.56\,\mathrm{m\,s^{-2}}$	0.10	0.098	0.048	0.064	**0.046**
5	$4.12\,\mathrm{m\,s^{-2}}$	0.091	0.077	0.046	0.065	**0.042**
6	$2.53\,\mathrm{m\,s^{-2}}$	0.13	0.055	0.048	0.043	**0.029**
7	$3.89\,\mathrm{m\,s^{-2}}$	0.076	0.047	0.028	0.045	**0.025**
8	$1.23\,\mathrm{m\,s^{-2}}$	0.031	0.030	0.019	0.028	**0.016**
9	$3.27\,\mathrm{m\,s^{-2}}$	0.074	0.082	0.036	0.042	**0.035**
10	$6.61\,\mathrm{m\,s^{-2}}$	0.19	0.068	0.058	0.076	**0.055**
11	$8.22\,\mathrm{m\,s^{-2}}$	0.48	0.13	0.10	0.091	**0.079**

Table 5.7: MAE of the V_x (in $\mathrm{m\,s^{-1}}$) estimation by the different observers calculated for the different trajectories in the testing set.

Trajectory	$a_y^{\max}$	DBM	4WM	KN	GRU	Ours
1	$1.82\,\mathrm{m\,s^{-2}}$	0.020	0.019	0.047	0.017	**0.014**
2	$2.45\,\mathrm{m\,s^{-2}}$	0.022	0.025	0.049	0.018	**0.015**
3	$2.1\,\mathrm{m\,s^{-2}}$	0.049	0.038	0.055	0.014	**0.011**
4	$3.56\,\mathrm{m\,s^{-2}}$	0.023	0.038	0.061	0.016	**0.014**
5	$4.12\,\mathrm{m\,s^{-2}}$	0.028	0.035	0.050	0.022	**0.019**
6	$2.53\,\mathrm{m\,s^{-2}}$	0.04	0.022	0.024	0.011	**0.008**
7	$3.89\,\mathrm{m\,s^{-2}}$	0.022	0.027	0.049	0.016	**0.015**
8	$1.23\,\mathrm{m\,s^{-2}}$	0.015	0.020	0.057	0.0143	**0.0140**
9	$3.27\,\mathrm{m\,s^{-2}}$	0.021	0.020	0.048	0.021	**0.019**
10	$6.61\,\mathrm{m\,s^{-2}}$	0.067	0.044	0.098	0.032	**0.031**
11	$8.22\,\mathrm{m\,s^{-2}}$	0.18	0.10	0.10	0.068	**0.065**

Table 5.8: MAE of the V_y (in $\mathrm{m\,s^{-1}}$) estimation by the different observers calculated for the different trajectories in the testing set.

Trajectory	$a_y^{\max}$	DBM	4WM	KN	GRU	Ours
1	$1.82\,\mathrm{m\,s^{-2}}$	2.60	2.62	2.88	2.73	**2.57**
2	$2.45\,\mathrm{m\,s^{-2}}$	2.17	2.22	2.49	2.64	**2.05**
3	$2.1\,\mathrm{m\,s^{-2}}$	2.15	2.48	2.52	4.02	**2.11**
4	$3.56\,\mathrm{m\,s^{-2}}$	2.09	2.30	2.53	2.90	**1.81**
5	$4.12\,\mathrm{m\,s^{-2}}$	2.60	2.88	3.06	3.25	**2.28**
6	$2.53\,\mathrm{m\,s^{-2}}$	3.09	3.96	3.61	2.97	**2.68**
7	$3.89\,\mathrm{m\,s^{-2}}$	2.08	2.57	2.52	2.42	**1.76**
8	$1.23\,\mathrm{m\,s^{-2}}$	1.40	1.65	1.97	2.0	1.40
9	$3.27\,\mathrm{m\,s^{-2}}$	2.34	3.37	2.87	2.65	**2.22**
10	$6.61\,\mathrm{m\,s^{-2}}$	6.02	8.40	7.06	6.32	**5.68**
11	$8.22\,\mathrm{m\,s^{-2}}$	15.8	17.0	18.5	16.1	**9.2**

Table 5.9: MAE of the $\dot{\psi}$ (in $\mathrm{mrad\,s^{-1}}$) estimation by the different observers calculated for the different trajectories in the testing set.

5.4.2.5 Can an attention-based observer easily substitute the proposed solution?

Having proven the accuracy of the proposed observer in the different considered scenarios, a question that arises is the ability of an attention-based observer to replace the LSTMs in the proposed architecture and to have similar or more accurate performance. To address this question, an encoder-only transformer-based architecture is tested. The architecture is inspired by the mass estimation observer proposed in [Zhang et al. 2023] and the part replacing the LSTMs of

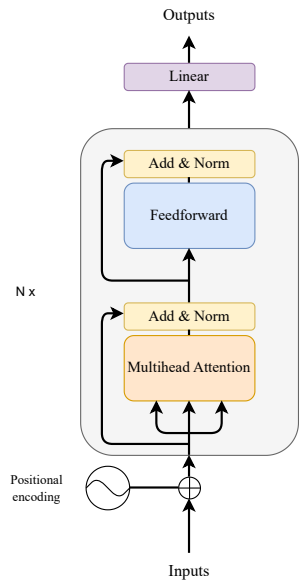


Figure 5.13: Attention-based observing architecture.

Figure 5.7 is shown in Figure 5.13 following the explanations provided in chapter 4 with $N = 3$ layers, 50 time steps, and 4 heads.

In comparison with the LSTM-based observer proposed earlier, the overall performance of the transformer-based method shows a V_x MAE of $0.22\,\mathrm{m\,s^{-1}}$, a V_y MAE of $0.026\,\mathrm{m\,s^{-1}}$ and a $\dot{\psi}$ MAE of $5.12\,\mathrm{mrad\,s^{-1}}$. Although the performance of the trained transformer model was not able to compete with other observers, it should be noted that changing the architecture, adding parameters, and performing other modifications may lead to better performance.

The use of transformers in observing applications was not investigated in the literature. A thorough study of transformer-based observers that includes analysis and comparisons with available solutions would be needed, which is out of the scope of this thesis.

5.4.2.6 Discussion

In brief, the proposed learning-based observer is able to accurately estimate the longitudinal and lateral velocities and yaw rate for all the considered scenarios while adapting to high dynamic cases. It is clear that model-based methods lose accuracy with higher dynamics due to several factors mentioned previously. The considered state-of-the-art methods are not able to reach the accuracy provided by the proposed method.

Next, we present another observing approach relying on a hybrid method to

deliver side-slip angle estimations.

5.5 Vehicle side-slip angle observer

The second observing application applied to the Stadtpilot vehicle is a side-slip angle observer. As seen in chapter 3, the side-slip angle of the vehicle is the angle between the velocity vector and the longitudinal axis at the center of gravity. Although it is considered as a critical quantity to assess vehicle stability, it is not accessible through standard sensors which made it a subject of research in the literature as mentioned in chapters 3 and 4. The aim of this Section is to accurately estimate it while adapting to high dynamic maneuvers.

To estimate the side-slip angle of the vehicle, a hybrid approach is proposed. The proposed approach makes use of a kinematic bicycle model (defined in chapter 2) and multilayer perceptrons. The neural network architecture aims to correct the errors of the kinematic side-slip to deliver accurate estimations.

In the following, the proposed architecture is presented in Section 5.5.1 and the trained observer is evaluated in Section 5.5.2. As it is the case for the previous observer, the proposed side-slip observer will be compared to state-of-the-art observers as well in low dynamic and high dynamic scenarios.

5.5.1 Proposed architecture

The aim of the proposed architecture is to deliver accurate side-slip angle estimations using only the standard in-car sensors. The iTraceRT measurements are once again used as reference.

The proposed architecture makes use of the kinematic side-slip angle defined in Equation (2.24d), chapter 2. The motivation is two-fold: on the one hand, the kinematic side-slip is able to capture, in many cases, the shape of the actual side-slip with offsets and errors especially in higher dynamics; on the other hand, using a physical model in addition to neural networks increases the physical interpretability of the solution.

The proposed architecture is shown in Figure 5.14. The inputs to the hybrid observer are the vehicle's standard sensors at a first stage and the kinematic side-slip angle at a second stage, both at the current time $k = t$. The reason behind this decision is that the network is expected to predict the corrections to be made to the kinematic side-slip from the given measurements instead of predicting the absolute side-slip angle value from the beginning (if inputs are all fed at the first stage); also, feeding all the inputs to the network at a first stage resulted in higher testing errors (up to 18%).

The multilayer perceptron includes in its first stage four layers with 16, 32, 64 and 128 neurons respectively. The output of the fourth layer is concatenated

with the kinematic side-slip angle and the result is fed, in a second stage, to two layers of 32 and 16 neurons respectively.

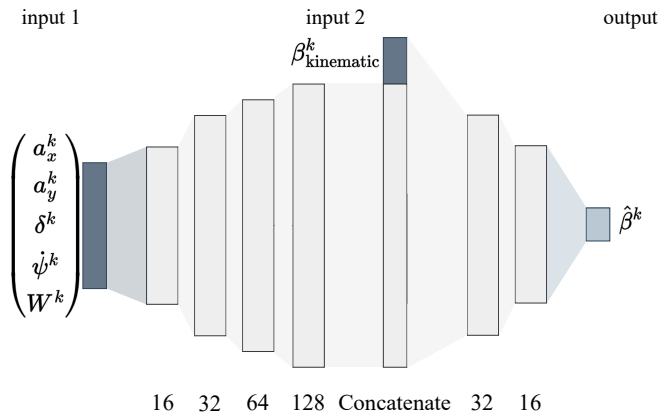


Figure 5.14: Side-slip angle observer architecture. $(k = t)$

All the activation functions are hyperbolic tangent (tanh) except for a linear activation function at the output layer. The size of the layers was determined using grid search. The model weights are initialized using the Xavier initialization. The loss function used is defined by:

$$L = L_\beta + L_{\text{reg}} \tag{5.1}$$

with L_β being an L2 loss comparing the output of the network and ground truth values, and L_{reg} being an L2 regularization for the network with a regularization rate of 10^{-5}. Regularization is used to avoid overfitting the network to the training data. It adds a penalty to the L2 norm of the weights.

The network is implemented in PyTorch and is trained using Adam on an Nvidia Geforce GTX 1650 Ti. Early stopping stops the training after 31 epochs.

After training the above-specified network, it is tested next.

5.5.2 Results

The trained observer is tested in this Section. A comparison with state-of-the-art approaches will give insight about the performance of the proposed approach. Two state-of-the-art approaches presented in chapters 3 and 4 are compared to the proposed approach, these are discussed in Section 5.5.2.1. The performance of all the observers is then evaluated for the whole testing set first, then for specific scenarios as performed in the previous section.

5.5.2.1 State-of-the-art observers

Two state-of-the-art model-based and learning-based techniques are used for comparison purposes with the proposed approach.

The chosen model-based observer involves an EKF based on a dynamic bicycle model with an adaptive linear tire [Reina and Messina 2019] referred to as DBM. The use of this observer will allow the analysis of using simplified model-based vehicle models for side-slip angle estimation. The observer is expected to lose accuracy with high dynamics. The considered model-based observer was implemented using the equations of the dynamic bicycle model provided in Chapter 2 and the equations of the EKF provided in Chapter 3. The model parameters used are communicated by the host laboratory. This does not eliminate inaccuracies, especially in the tire parameters, which have effects on the accuracy of the observer as seen later on.

The chosen learning-based observer is a hybrid one [Graber et al. 2019] that uses a GRU network with a dynamic bicycle side-slip rate input to deliver side-slip angle estimations. The side-slip rate is calculated using the standard in-car sensor measurements. It will be referred to as GRU. The mentioned work compares to a fully learned approach with no physical model input and is able to get better results; for this reason, we do not consider a comparison with a fully learned approach below. The considered hybrid observer was implemented using PyTorch and was trained on the same training data used to train our approach.

Remark 15. Proving the ability of the proposed network to outperform the hybrid work done in [Graber et al. 2019] will demonstrate that the job can be done without considering temporal dependencies, and with the use of the physical input of a simpler model (the kinematic model).

5.5.2.2 Overall performance

The metric used to compare the different approaches is the mean absolute error (MAE). An evaluation procedure similar to the one used in the previous velocity observer is employed here by evaluating the whole testing set first, then moving to specific low dynamic and high dynamic maneuvers.

The overall performance of the different observers on the whole testing set is shown through the calculation of MAE with respect to the reference in Table 5.10. The table shows that the proposed approach delivers the lowest errors. The GRU approach is better than the DBM approach but shows almost 1.4 times higher errors than the proposed approach.

State	DBM	GRU	Ours
β (mrad)	4.27	3.25	**2.37**

Table 5.10: Mean absolute error (MAE) for the different observers calculated for the whole testing set. The errors of our approach are the lowest among the state-of-the-art observers.

To further inspect the performance of the proposed method we split the analysis into low and high dynamic maneuvers in the next sections. The same high dynamic maneuver shown in Figure 5.8 is used, but a maneuver with $a_y^{\max} = 0.35g$ collected also during inner city driving replaces the low dynamic scenario to better visualize the performance of the observers due to minimal side-slip angle variations in the former maneuver.

5.5.2.3 Low dynamic maneuver

The evaluation begins with evaluating the performance of the observers in a low dynamic maneuver. The different MAE are calculated for the different observers and are shown in Table 5.11. The table shows that the proposed observer is able to beat the considered state-of-the-art approaches by a factor of 1.3 to 2.2 on average.

State	DBM	GRU	Ours
β (mrad)	3.98	2.35	**1.76**

Table 5.11: MAE for the different observers calculated for the low dynamic driving maneuver. The errors of our approach are the lowest among the state-of-the-art observers.

To closely visualize the performance of the different approaches, we plot in Figure 5.15 the side-slip angle estimations along with the reference sensor values for a sequence in the highest lateral acceleration region of the considered trajectory. The plot shows the capability of the proposed method to closely follow the values of the reference iTraceRT sensor. The DBM method seems accurate in the rising part of the curve while inaccuracies can be seen at the peak and towards the decreasing parts of the curve. The GRU undershoots through most of the curve.

To investigate further the behavior of the observers, a high dynamic scenario is considered next.

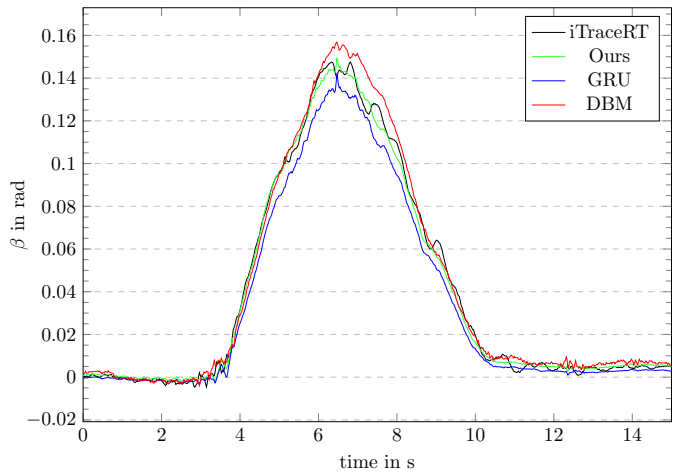


Figure 5.15: Performance of the different observers for the low dynamic maneuver. The proposed method is able to deliver the most accurate estimations.

5.5.2.4 High dynamic maneuver

A high dynamic scenario is defined, as in the previous section, as a one having high lateral accelerations surpassing 0.5 g.

The MAE of the different observers for the high dynamic scenario are shown in Table 5.12. All the observers present errors higher than the low dynamic case. Though, the table shows that the proposed approach is able to deliver the lowest errors.

A visualization of a sequence in the highest lateral acceleration region in Figure 5.16 shows the advantage of the proposed approach in sticking to the reference iTraceRT values. The DBM approach fails to deliver accurate results: this behavior is expected from a dynamic bicycle based observer for model validity reasons.

Remark 16. The high error shown by the dynamic bicycle based observer can be attributed to the linear property of the tire model, which is the result of the inability of the EKF to update accordingly the cornering stiffness values.

State	DBM	GRU	Ours
β (mrad)	9.79	7.48	**4.42**

Table 5.12: MAE for the different observers calculated for the high dynamic maneuver. The errors of our approach are the lowest among the state-of-the-art observers.

To inspect the behavior of each of the observers with respect to the harshness of the maneuver, the absolute errors with respect to the reference of each of the

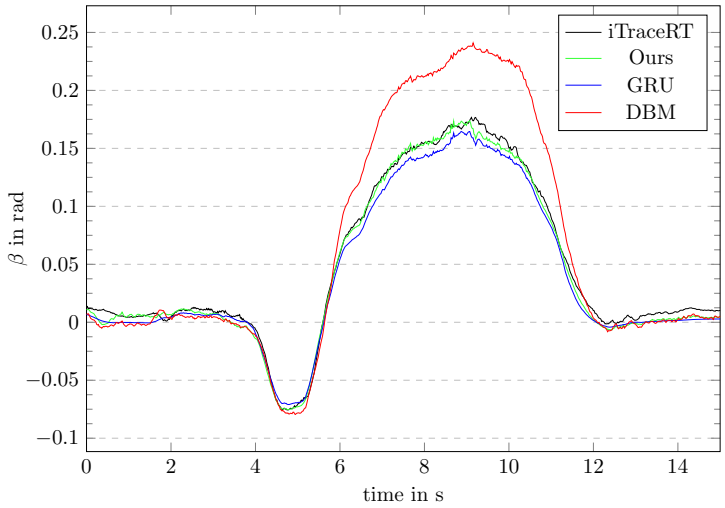


Figure 5.16: Performance of the different observers for the high dynamic maneuver. The proposed method is able to deliver the most accurate estimations.

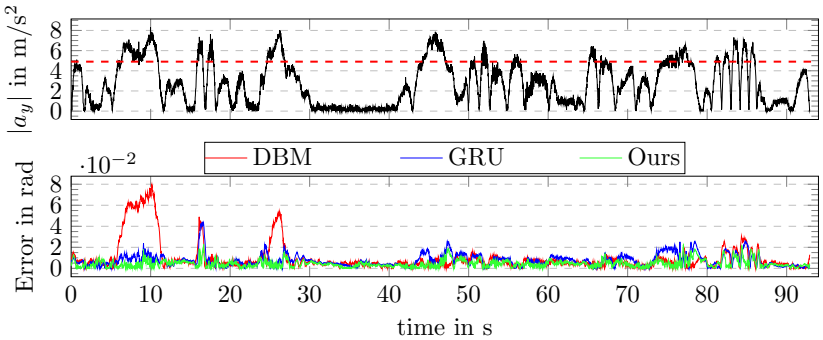


Figure 5.17: Errors of the different observers compared to the evolution of the lateral accelerations along the harsh maneuver. The proposed approach shows the lowest errors while the EKF based on the dynamic bicycle model shows the highest errors.

observers are plotted in parallel to the evolution of the lateral acceleration of the vehicle in Figure 5.17. The plot shows that the proposed approach presents the lowest errors. The errors of all the observers increase with the rise of the lateral acceleration. The DBM approach shows high sensitivity to high lateral accelerations, its errors go up to 0.081 rad; the GRU approach performs better but its errors can reach that of the DBM approach (e.g. at t = 16 s); the proposed observer is robust to high accelerations and maintains the lowest errors along the trajectory.

Note that the errors of all the testing trajectories are shown in Table 5.13.

Trajectory	$a_y^{\max}$	DBM	GRU	Ours
1	$1.82\,\mathrm{m\,s^{-2}}$	2.5	2.9	**2.17**
2	$2.45\,\mathrm{m\,s^{-2}}$	2.40	2.44	**2.11**
3	$2.1\,\mathrm{m\,s^{-2}}$	2.19	2.29	**1.43**
4	$3.56\,\mathrm{m\,s^{-2}}$	3.98	2.35	**1.76**
5	$4.12\,\mathrm{m\,s^{-2}}$	2.67	2.60	**2.11**
6	$2.53\,\mathrm{m\,s^{-2}}$	3.96	3.23	**2.25**
7	$3.89\,\mathrm{m\,s^{-2}}$	1.90	1.95	**1.57**
8	$1.23\,\mathrm{m\,s^{-2}}$	1.51	2.47	**1.28**
9	$3.27\,\mathrm{m\,s^{-2}}$	2.51	3.43	**2.18**
10	$6.61\,\mathrm{m\,s^{-2}}$	4.3	2.87	**2.19**
11	$8.22\,\mathrm{m\,s^{-2}}$	9.79	7.48	**4.42**

Table 5.13: MAE of the side-slip angle (in mrad) estimation by the different observers calculated for the different trajectories in the testing set.

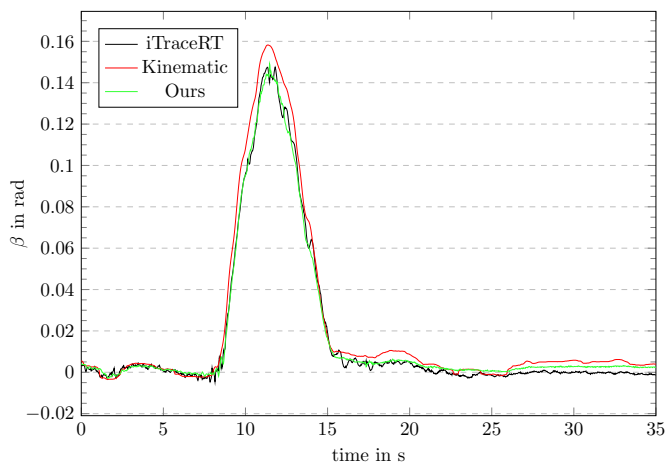


Figure 5.18: Comparison between the kinematic side-slip angle, the estimations of the proposed observer and the reference iTraceRT values. The plot shows the ability of the proposed approach to adjust the kinematic input and provide estimations closer to the ground truth.

After proving that the proposed method is able to outperform the considered state-of-the-art methods, we inspect the difference between the kinematic side-slip angle, the estimations of the proposed network and the ground truth values. We plot a sample side-slip angle rise in Figure 5.18. The figure shows that the proposed approach is able to correct the kinematic side-slip values and provide

estimations closer to the ground truth. The proposed approach takes advantage of the kinematic side-slip angle but is able to adapt to its errors to deliver accurate outputs.

5.5.2.5 Discussion

In brief, the proposed observer is able to outperform the considered state-of-the-art methods for all the considered scenarios.

In comparison with the considered model-based observer, the proposed observer showed more robustness to higher accelerations. The DBM approach presented significant errors which is logical due to model validity issues in harsh maneuvers.

In comparison with the observer presented in [Graber et al. 2019], it is able to provide more accurate estimations using a simpler physical model and a simpler neural network architecture, which makes it superior in multiple aspects.

5.6 Conclusion

In this chapter, learned observers were applied to real vehicles in two applications: longitudinal and lateral velocity and yaw rate estimation, and side-slip angle estimation. Each of the proposed observers has its unique architecture that takes advantage of the provided vehicle measurements to deliver accurate estimations. The two proposed observers were compared to state-of-the-art model-based and learning-based observers. The limitations of different model-based observers were explored in different scenarios and their inability to provide accurate estimations was shown especially in harsh maneuvers.

The proposed methods were able to outperform state-of-the-art methods in both applications and were able to adapt to high dynamic maneuvers delivering accurate estimations even at the limits of handling.

At this point, integrating learning techniques to observe the state of the vehicle showed many advantages, reaching the ability to accurately estimate the velocities and the side-slip angle even for high dynamics. For the remaining part of this thesis, the already targeted variables of the state of the vehicle are then assumed to be accessible. Having accurate state knowledge allows proceeding to plan vehicle trajectories. This is targeted next.

Part III

Motion Planners

Chapter 6

HEBM: A Hybrid Model for Accurate Trajectory Planning

Contents

6.1 Introduction **112**
6.2 Overview **112**
6.2.1 Model predictive control (MPC) 113
6.2.2 Model predictive path integral (MPPI) 114
6.3 The Hybrid Extended Bicycle **116**
6.3.1 Data generation 117
6.3.2 Methodology 118
6.4 Proposed approach **120**
6.4.1 MPPI architecture 120
6.4.2 Low level controllers 121
6.5 Results **123**
6.5.1 Metric 123
6.5.2 Oval trajectory 123
6.5.3 Lane change trajectory 127
6.5.4 Discussion 129
6.6 Conclusion **129**

6.1 Introduction

As it is clear by now, planning trajectories to perform driving maneuvers based on erroneous vehicle knowledge results in inaccurate vehicle behavior. After developing learning-based observers to estimate accurately multiple variables representing the state of the vehicle, these variables were assumed accessible at the end of the previous part. Given this assumption, a new challenge is addressed in this chapter: planning in high dynamics.

In autonomous driving applications, the planning layer makes use of a vehicle model to decide on the optimal trajectory to be followed to complete a given task. The validity of the used model is an essential property guaranteeing the feasibility of the planned trajectory. As seen in the previous chapters, difficulties are present when describing the behavior of the vehicle in high dynamics. This is due to the high nonlinearities (especially at the tire level) defying the hypotheses (check chapter 3) used for normal driving. In other words, the linear tire behavior and no-slip condition employed in simple vehicle models are no longer valid. The validity of the used model to make decisions highly impacts the behavior of the vehicle and therefore affects its safety.

This chapter focuses on augmenting the extended bicycle model, an extension to the kinematic bicycle model already used in the literature for different applications, with recurrent neural networks to reach a simple hybrid model with a larger domain of validity even when high dynamic maneuvers are involved. The aim is to be able to plan accurate trajectories despite the harshness of the encountered scenario. This is the last contribution of this thesis.

In the following sections, a brief overview of planning methods is presented while stating the targeted issue, the proposed approach is then presented and tested on several maneuvers.

6.2 Overview

Solving the motion planning problem, which mainly consists of reaching a final state given the initial state and the conditions of the environment, relies on different methods [Gonzalez et al. 2016] that could be split into four groups: graph search based planners, sampling-based planners, interpolating curve planners and numerical optimization approaches. Each of the approaches is briefly introduced below while focusing on optimization approaches for the reasons stated in the corresponding part.

Graph search based planners include algorithms such as A* [Hart et al. 1968] or Dijkstra [Dijkstra 1959] that assume a decomposed configuration space (a configuration space is a space containing all the possible configurations of a system) in which the algorithm could be employed to find the minimal cost between two

configurations. Example usage includes planning using the Dijkstra algorithm in the DARPA[1] challenge [Bohren et al.; Bacha et al. 2008; 2008] or using the A* algorithm for planning in parking lots [Ziegler and Werling 2008].

Deterministic sampling-based planners consist in decomposing the configuration space into cells using different methods [LaValle 2006] then applying graph search based planners to get the optimal solution between two configurations. Probabilistic sampling-based planners are employed in high dimensional configuration spaces where the deterministic graph search based planners become computationally expensive. Probabilistic sampling-based methods such as Rapidly-exploring Random Trees (RRT) [LaValle and Kuffner 2001] randomly generate new points in the configuration space looking for potential solutions to the planning problem. RRT was used for example by the MIT team in the DARPA challenge [Kuwata et al. 2011].

Interpolating curve planners are used to construct a new (smoother) set of points given a previously known set. This allows trajectory continuity and dealing with a dynamic environment. Methods include using lines and circles, clothoid curves, polynomial curves, Bézier curves, and spline curves. Example implementations can be seen in [Ming Feng Hsieh and Ozguner; Broggi et al. 2008; 2012].

Optimization techniques including the use of optimal control approaches to solve the planning problem have been popular in autonomous driving applications during the last decade. These methods are able to integrate several constraints and include vehicle dynamics to result in a generated reference trajectory instead of just a reference path. This makes optimization techniques of interest in recent vehicle planning works and in our work that intends to use a model describing the vehicle to ensure feasible trajectory planning. For this purpose, the model predictive control (MPC) and the model predictive path integral (MPPI) optimization-based methods are presented next.

6.2.1 Model predictive control (MPC)

Model predictive control [Camacho et al. 2003] is a very popular control technique that computes the optimal control sequence that minimizes a predefined cost function over a predefined horizon.

The problem considers a system whose state Z is subject to the update function imposed by the system's dynamics. Constraints are added to the system's states and controls.

The quadratic cost function J to be minimized depends on the state Z of the system and its controls U with corresponding weight matrices Q, R:

$$J(Z_t, U_t) = \sum_{i=0}^{N_y} Z_t^{i\top} Q Z_t^i + \sum_{j=0}^{N_u-1} U_t^{j\top} R U_t^j \tag{6.1}$$

[1] https://www.darpa.mil/about-us/timeline/-grand-challenge-for-autonomous-vehicles

with N_y being the number of state predictions ($N_y = \frac{T_y}{\Delta t_y} + 1$, T_y is the state prediction horizon and Δt_y is the prediction time step); and N_u being the number of control steps ($N_u = \frac{T_u}{\Delta t_u} + 1$, T_u is the control prediction horizon and Δt_u is the control time step); such that $N_u \leq N_y$.

At each time step, the MPC algorithm would solve the above-defined optimization problem to deliver the optimal control sequence for the defined horizons, the first element (or set of elements) of the optimal control sequence is then applied to the system before launching the algorithm again. In planning applications, the MPC algorithm would be employed at the planning level to deliver reference high-level controls for the low-level controllers to follow based on the defined cost function. Example implementations include using the kinematic bicycle model [Cardoso et al.; Abbas et al.; Polack et al.; Burger et al. 2016; 2017; 2018; 2022] or the dynamic bicycle model [Park et al. 2009] to plan vehicle trajectories. Many variants of the MPC algorithm are present in the literature, they are not within the scope of this thesis.

Another optimal control approach is the model predictive path integral presented next.

6.2.2 Model predictive path integral (MPPI)

The Model Predictive Path Integral (MPPI) [Williams et al.; Williams et al. 2016; 2018] is a sampling-based, derivative-free, model predictive control algorithm. The approach consists of making use of the Graphics Processing Unit (GPU) to sample a large number of trajectories based on a given model. The evaluation of the generated trajectories will lead to the computation of the optimal control sequence.

Algorithm 1 describes the operation of the MPPI approach.

The algorithm starts by defining the number of samples, which is the number of trajectories τ to be generated, each having N time steps. The different trajectories are generated based on a model whose evolution is described by a function f to which inputs $u + \delta u$ are applied. Z_{t_0} denotes the initial state of the model. δu is sampled from a uniform distribution and added to an initial control sequence updated at each iteration.

The cost function S involves a running cost term $\hat{q}$ computed at each time step and a terminal cost term ϕ computed at the end of the generation process, $\hat{q}$ being:

$$\hat{q} = q(Z_t) + \frac{1 - \nu^{-1}}{2} \delta u_t^\top R \delta u_t + u_t^\top R \delta u_t + \frac{1}{2} u_t^\top R u_t \tag{6.2}$$

ν being the exploration noise which determines how aggressively MPPI explores the state space and R being a positive definite control weight matrix. The control sequence is then updated according to the cost of each trajectory while taking into consideration the minimum cost and the inverse temperature λ that impacts

the degree of selectiveness of the algorithm. The updated control sequence is smoothed using the Savitzky-Golay [Savitzky and Golay 1964] convolutional filter. The first control of the sequence is returned and the process reiterates.

As presented in [Mohamed et al. 2021], Figure 6.1 shows an illustration of the operation of the algorithm. Note that in planning applications the order may not be the same as shown in the Figure because of the difference in frequency between the planner and the low level controller.

Algorithm 1 MPPI Algorithm

Given: K : Number of Samples;
N : Number of time steps;
$(u_0, u_1, ...u_{N-1})$: Initial control sequence;
$f, \Delta T$: Dynamics function, step size;
ϕ, q, λ: Cost function/Hyperparameters;
SGF: Savitzky-Golay (SG) convolutional filter;
while *task not completed* **do**
  $\delta u \leftarrow$ RandomNoiseGenerator();
  $\hat{S}(\tau_k) \leftarrow$ CostInitializer();
  for $k \leftarrow 0$ **to** $K-1$ **do**
    $Z \leftarrow Z_{t_0}$;
    for $t \leftarrow 1$ **to** N **do**
      $Z_{t+1} \leftarrow Z_t + f(Z_t, u_t + \delta u_{t,k})\Delta T$;
      $\hat{S}(\tau_{t+1,k}) \leftarrow \hat{S}(\tau_{t,k}) + \hat{q}$;
    end for
    $\hat{S}(\tau_k) \leftarrow \hat{S}(\tau_{N,k}) + \phi(z_N)$;
  end for
  $\hat{S}_{\min} \leftarrow \min_k[\hat{S}(\tau_k)]$;
  for $t \leftarrow 0$ **to** $N-1$ **do**
    $u_t \leftarrow u_t + \frac{\sum_{k=1}^{K} \exp\left(-(1/\lambda)[\hat{S}(\tau_k) - \hat{S}_{\min}]\right)\delta u_{t,k}}{\sum_{k=1}^{K} \exp\left(-(1/\lambda)[\hat{S}(\tau_k) - \hat{S}_{\min}]\right)}$;
  end for
  $u \leftarrow$ SGF(u);
  Apply(u_0);
  for $t \leftarrow 1$ **to** $N-1$ **do**
    $u_{t-1} \leftarrow u_t$;
  end for
  $u_{N-1} \leftarrow$ Initialize(u_{N-1});
end while

The advantage of the approach is its ability to easily integrate learned dynamics into the optimization process while integrating learned models into classic optimization tools (e.g. MPC) would be more complex and computationally de-

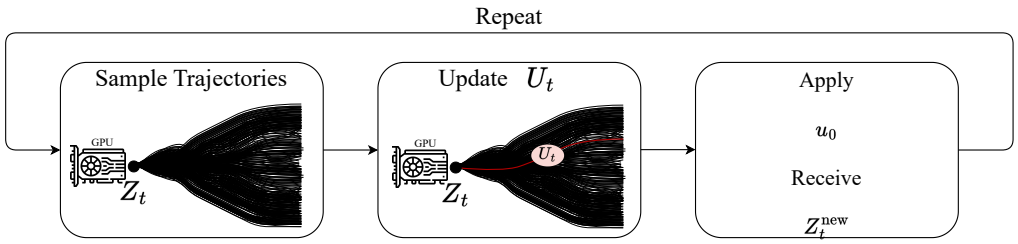


Figure 6.1: MPPI algorithm.

manding due to the complexity of the equations of the neural networks, especially if time dependency is included.

While the presented MPC and MPPI approaches integrate the dynamics of a vehicle model to be able to plan trajectories, the validity of the used model is necessary to avoid planning infeasible trajectories. In addition to the validity of the model used, its simplicity is an important fact in a planner that is supposed to make fast decisions. Thus the motivation to develop the hybrid extended bicycle model based planner presented and tested in the remaining parts of this chapter.

6.3 The Hybrid Extended Bicycle

The aim of this chapter is to introduce a simple hybrid model able to represent the behavior of the vehicle even in harsh maneuvers. The model is used later-on in planning vehicle trajectories. Therefore, the extended bicycle model is used.

The Extended Bicycle Model (EBM) was presented in chapter 2. As a reminder, the EBM is the one shown in Figure 6.2, extending the kinematic bicycle by the slip angles at the wheels. The state evolution of the model is described in Equations (2.23).

The extended bicycle model introduces accurate properties through the inclusion of the slip angles at the wheels, the difficulty remains in identifying the slip angles without referring to the vehicle's dynamics. The proposed approach solves this problem by using an LSTM-based slip angle predictor. The combination of the extended bicycle model with the slip angle predictor will be referred to as the Hybrid extended bicycle model (HEBM).

To be able to train the proposed slip angle predictor, a training dataset should be created. The four-wheel vehicle model with a Pacejka tire model presented in chapter 2 is assumed to be the reference in this work. In other words, the introduced solution should be able to represent as much as possible the behavior of the four-wheel model. The training data generation is presented next, followed by the proposed methodology and architecture.

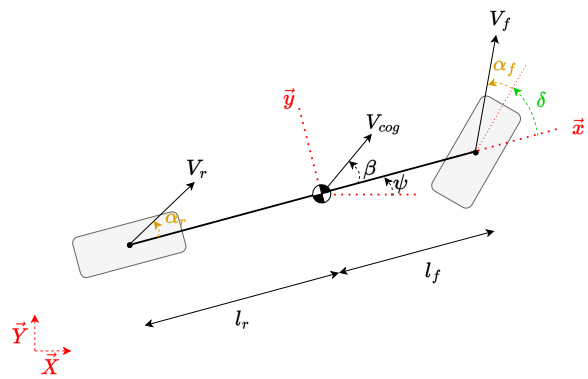


Figure 6.2: The extended bicycle model.

6.3.1 Data generation

As mentioned earlier, the aim is to develop a hybrid extended bicycle able to represent the behavior of the vehicle depicted by a four-wheel model in our case. Therefore, the reference four-wheel model is used for data generation.

The training dataset is generated by applying control inputs (torque and steering angle) to the four-wheel model. The dataset is made of 5000 2-second trajectories sampled at 100 Hz leading to a total of 1 million samples. The procedure used to create the dataset is the following:

1. A random initial vehicle state is chosen.

2. Random controls are drawn from a uniform distribution as follows:

   - A torque distribution of $T \in [0, 800\,\mathrm{N\,m}]$ if $V < 10\,\mathrm{m\,s^{-1}}$, $T \in [-1000\,\mathrm{N\,m}, 800\,\mathrm{N\,m}]$ if $10\,\mathrm{m\,s^{-1}} < V < 30\,\mathrm{m\,s^{-1}}$ and $T \in [-1000\,\mathrm{N\,m}, 0]$ if $V > 30\,\mathrm{m\,s^{-1}}$.
   - A steering angle distribution of $\delta \in [-0.5\,\mathrm{rad}, 0.5\,\mathrm{rad}]$.
   - A constant control period is uniformly drawn between 0.01 s and 1 s.
   - Five control samples are considered and the one that results in a more diverse distribution on the friction circle is chosen.

3. The sampled controls are applied to the four-wheel model and are held constant for the constant control period. The state of the reference model is updated accordingly.

4. The procedure repeats from Step (2) until a 2-second trajectory is formed.

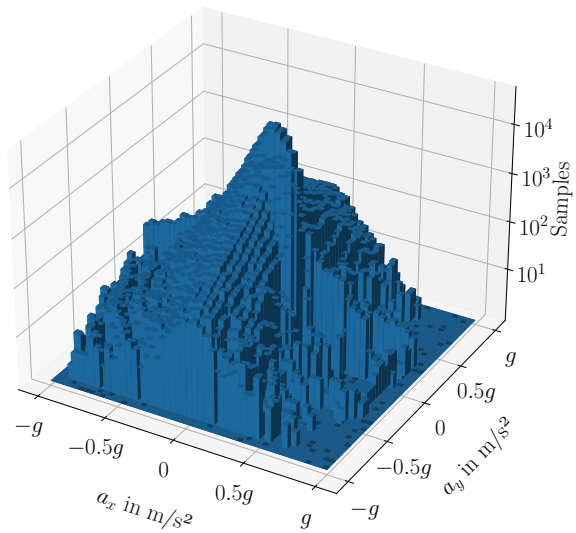


Figure 6.3: Distribution of the generated set on the friction circle (number of samples shown on a logarithmic axis). The plot shows a distribution containing low and high acceleration maneuvers reaching ($a > 0.5g$).

The resulting distribution of the data samples on the friction circle is shown in Figure 6.3. The plot shows more samples in the low acceleration areas but harsh maneuvers are present with accelerations reaching $|a| > 0.5g$.

The created dataset is split into training and validation datasets (70%-30%). The training of the network using the generated datasets is defined next.

6.3.2 Methodology

As the extended bicycle model lacks the knowledge of the slip angles at the front and rear wheels to be able to operate, a slip angle predictor is introduced. The proposed predictor makes use of LSTM networks defined in chapter 4 exploiting the temporal dependencies between the consecutive states and controls of the model to predict the wheel slip angles.

The proposed architecture is shown in Figure 6.4. As shown in the figure, to be able to predict the slips at $t = k$ the inputs to the network combine the vehicle state part that involves the longitudinal and lateral velocities and yaw rate for the last 10 time steps ($t = k - 1..k - 10$), and the controls part that involves the controls (the velocity V and the steering angle δ) applied at the last 9 time steps and the ones to be applied to reach the state at the current time step ($t = k..k-9$). The outputs of the network are the resulting slip angles at the wheels at $t = k$

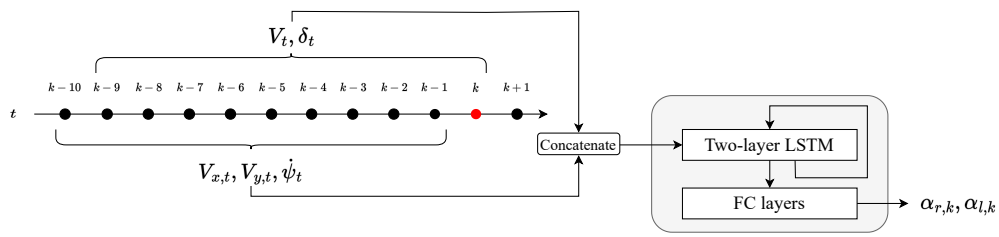


Figure 6.4: Slip angle predictor architecture.

when the considered controls are applied. The output of the network will allow the computation of the state evolution of the model from $t = k-1$ to $t = k$ using Equations (2.23).

At each time step the proposed network will provide the slip angles necessary to calculate the evolution of the state of the extended bicycle model. The architecture and training details are presented next.

6.3.2.1 Architecture definition and training

The network shown in Figure 6.4 is made of two LSTM layers consisting of 32 and 64 neurons respectively and three fully connected layers of 128, 256 and 128 neurons respectively. ReLU activation functions are used except for linear activation functions at the output layer.

The loss function used makes use of the predicted slips to compute the state evolution of the model and then compares the resulting longitudinal and lateral velocities and yaw rate to the reference data. In other words, the training process starts by a forward pass through the network resulting in the slip angles outputs: α_f, α_r; the output will be used to calculate the side-slip angle of the model using Equation (2.23d); the state evolution of the extended bicycle model is then calculated using Equations (2.23a)-(2.23c), the velocities are projected to the vehicle frame using the following equations to allow the comparison with the reference:

$$V_x = \dot{Y}_{cog} \sin\psi + \dot{X}_{cog} \cos\psi \tag{6.3a}$$

$$V_y = \dot{Y}_{cog} \cos\psi - \dot{X}_{cog} \sin\psi \tag{6.3b}$$

The velocities and the yaw rate are compared to the reference through an L1 loss that will be back-propagated through the network. The loss function used will emphasize on lateral dynamics as the tests showed the ease of learning the longitudinal dynamics by the network in comparison with the lateral dynamics. Hence, the loss function used is defined as follows:

$$L = 0.2L_1^{V_x} + 0.4L_1^{V_y} + 0.4\frac{1}{\gamma}L_1^{\dot{\psi}} \tag{6.4}$$

with $\gamma = 0.05$ being a scaling factor to account for the scaling difference between the velocities V_x, V_y and the yaw rate $\dot{\psi}$.

The defined architecture is implemented and trained using PyTorch on an Nvidia Geforce GTX 1650 Ti for 42 epochs (early stopping is used).

Having an extended model with a trained LSTM network for slip angle prediction, the hybrid model can be employed in planning applications as presented next.

6.4 Proposed approach

The introduced hybrid extended model was trained to provide a behavior as close as possible to the reference. Thus, the dynamics of the system to be used to plan vehicle trajectories is a combination of deterministic equations and recurrent neural networks. For this reason, the presented solution will make use of the MPPI approach introduced previously to plan feasible trajectories.

In the proposed planner, the reference path and reference speeds are assumed to be known. The aim of the proposed solution is to deliver the reference velocity V_{ref} and steering angle δ_{ref} for the vehicle to follow to be able to be as close as possible to the reference path and reference speed even if the vehicle has to engage in high dynamic maneuvers. The block diagram depicting the operation of the proposed method is shown in Figure 6.5 and its details are presented in the next sections with the MPPI approach based on the hybrid extended bicycle model introduced first followed by the low-level controllers employed to effect the planned trajectories.

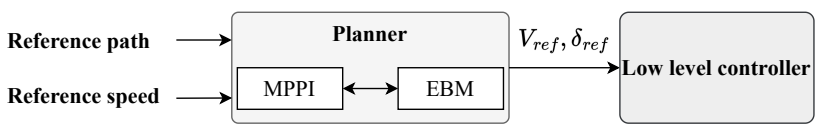


Figure 6.5: Planner-Controller proposed architecture.

6.4.1 MPPI architecture

The proposed planner makes use of Algorithm 1 detailed in the previous section. In the proposed planner, the function f depicting the state evolution of the system is the combination of Equations (2.23) and the slip predictor architecture mentioned previously. The system used then is the hybrid extended bicycle model. The used running cost q associated with the proposed MPPI planner is defined as:

$$q(Z) = (Z_t - Z^{\text{ref}})^\top Q_Z (Z_t - Z^{\text{ref}}) + (V_t - V^{\text{ref}})^\top Q_V (V_t - V^{\text{ref}}) \tag{6.5}$$

Z being the state of the extended bicycle model defined as $Z = \begin{bmatrix} X & Y & \psi \end{bmatrix}^\top$ which is compared to the reference path's X and Y coordinates and heading, Z^{ref}. The used terminal cost is $\phi(Z_N) = q(Z_N)$. δu is issued from a normal distribution between $-0.03\,\mathrm{m\,s^{-1}}$ and $0.05\,\mathrm{m\,s^{-1}}$ for the velocity V control and between $-0.02\,\mathrm{rad}$ and $0.02\,\mathrm{rad}$ for the steering angle δ control. The cost matrices are: $Q_Z = 4 * \text{ Diag}(1, 1, 10)$, $Q_V = 3$, $R = .1 * \text{ Diag}(1, 1)$.

The parameters defining the used MPPI method are shown in Table 6.1.

At each iteration, the proposed MPPI-based planner will use the current reference vehicle state to sample $K = 1024$ trajectories for $N = 100$ time steps using the hybrid extended model; the planner outputs the optimal control sequence.

As the planner will run at 20 Hz while the low-level controllers introduced next will run at 100 Hz, the first five elements of the optimal control sequence are sent to the low-level controllers.

Description	**Parameter**	**Value**
Inverse temperature	λ	$3e-01$
Exploration noise	ν	1000
Total number of samples	K	1024
Total number of time steps	N	100
Time step	ΔT	0.01 s

Table 6.1: MPPI Parameters

Remark 17. Other MPPI algorithms[2] developed in the Autonomous Control and Decision Systems Laboratory of the Georgia Institute of Technology could be used and may lead to better results.

6.4.2 Low level controllers

The reference four-wheel vehicle presented in chapter 2 takes as inputs the torques T applied at the wheels and the steering angle δ at the front wheels. As the proposed architecture suggests, the planner will deliver the reference velocities and steering angles that the vehicle should follow to result in optimal trajectories. Low level controllers are then required to link between the planner's output and control inputs of the four-wheel model.

In the proposed approach the low level control is split into longitudinal control aiming to control the torque of the vehicle to reach the reference velocity provided by the planner and lateral control aiming to control the steering angle to reach the reference steering angle provided by the planner. Each of the controllers is detailed next.

[2] https://sites.gatech.edu/acds/mppi/

6.4.2.1 Longitudinal controller

The longitudinal controller aims, by controlling the torque, to make the four-wheel vehicle model reach the reference velocity provided by the planner. The error to minimize is then defined as:

$$e_V(t) = V_{\text{ref}} - V_{\text{vehicle}}(t) \tag{6.6}$$

V_{ref} being the reference velocity provided by the planner and V_{vehicle} being the four-wheel vehicle velocity calculated as $V_{\text{vehicle}}(t) = \sqrt{V_x^2(t) + V_y^2(t)}$. The controller used is a simple PID [Åström and Hägglund 1995] controller defining the applied torque in function of the error as follows:

$$T(t) = K_{P,V} e_V(t) + K_{D,V} \dot{e}_V(t) + K_{I,V} \int_0^t e_V(\tau) d\tau \tag{6.7}$$

The mentioned gains are defined in Table 6.2. They were tuned manually.

6.4.2.2 Lateral controller

The lateral controller aims to control the four-wheel model's steering angle to reach the steering angles provided by the planner. The chosen lateral control approach splits the problem in two as done in [Polack et al. 2018]: an open loop part and a closed loop part. The open loop part is equal to the steering angle value delivered by the planner δ_{ref} while the closed loop part is a PID controller aiming to minimize the heading error e_ψ between the four-wheel vehicle and the projection of the hybrid extended bicycle model over the five time steps given the provided reference controls:

$$e_\psi(t) = \psi_{\text{HEBM, projected}}(t) - \psi_{\text{vehicle}}(t) \tag{6.8}$$

The applied steering angle is then defined as:

$$\delta(t) = \delta_{\text{ref}} + K_{P,\delta} e_\psi(t) + K_{D,\delta} \dot{e}_\psi(t) + K_{I,\delta} \int_0^t e_\psi(\tau) d\tau \tag{6.9}$$

The mentioned gains are tuned manually and are defined in Table 6.2.

$K_{P,V}$	$K_{D,V}$	$K_{I,V}$	$K_{P,\delta}$	$K_{D,\delta}$	$K_{I,\delta}$
500	20	15	0.2	0.01	0.005

Table 6.2: Longitudinal and lateral PID Parameters

Having defined the proposed planning and controlling approach, the system is tested next.

6.5 Results

The proposed planner-controller architecture is tested in this Section. The test evaluates the behavior of the four-wheel vehicle model which trajectories would be planned using the MPPI-HEBM structure and which torque and steering would be controlled using the low-level controllers presented before.

As mentioned before, the planner would take as inputs the reference path and speeds, which will vary between test cases. The four-wheel model is controlled accordingly. The approach is compared to the kinematic bicycle model (KBM) which is used, as mentioned previously, in various literature applications. For a fair comparison, the kinematic bicycle model will be implemented in a similar architecture as the one proposed earlier. The kinematic bicycle model would replace the HEBM shown in Figure 6.5.

The evaluation will be split into two: an oval trajectory test and a lane change trajectory test. In the first test, the vehicle will have to follow an oval path in a clockwise (CW) and counter-clockwise (CCW) manner at various speeds; this will allow the comparison between the behavior of both models in harsh turns in different directions. In the second test, the vehicle will have to follow a lane change path at various speeds; simulating the behavior of both approaches in a challenging real-life scenario.

In what follows, the metric used to evaluate the approaches is discussed before moving to the comparisons for the oval test and the lane change test.

6.5.1 Metric

To evaluate the performance of the approach, the mean absolute error (MAE) and the max absolute error are calculated for each test in addition to the mean velocity associated with the maneuver. The error is defined as the lateral error between the vehicle and the reference path:

$$e_{\text{lateral}} = (Y_{\text{vehicle}} - Y_{\text{ref}}) \cos \psi_{\text{ref}} - (X_{\text{vehicle}} - X_{\text{ref}}) \sin \psi_{\text{ref}} \tag{6.10}$$

X_{ref}, Y_{ref}, ψ_{ref} are associated with the closest reference points to the vehicle. The tests are presented next.

6.5.2 Oval trajectory

The oval trajectory test consists of running the vehicle on the reference path shown in Figure 6.6 at varying speeds starting from $X = 0$, $Y = 0$ with $V_{x,0} = V_{\text{ref}}$, the other states being null except for the wheel speeds that are calculated accordingly with no initial slip. The corresponding curvature is shown in Figure 6.7. The results of the effected test, along with the reference speeds are shown in Table 6.3.

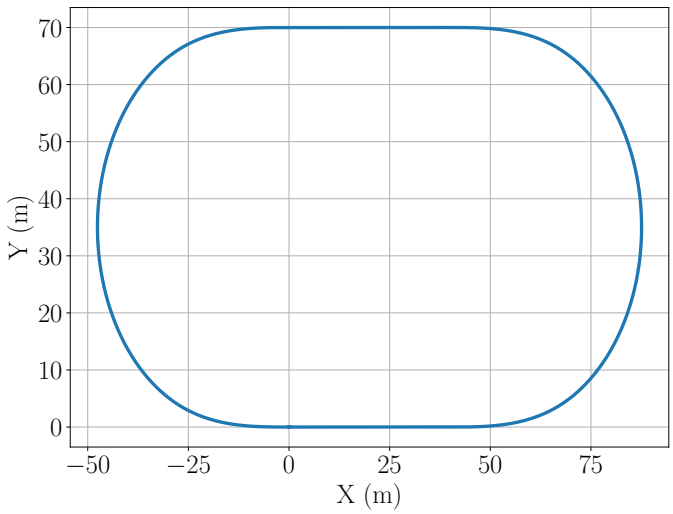


Figure 6.6: Oval trajectory test reference path.

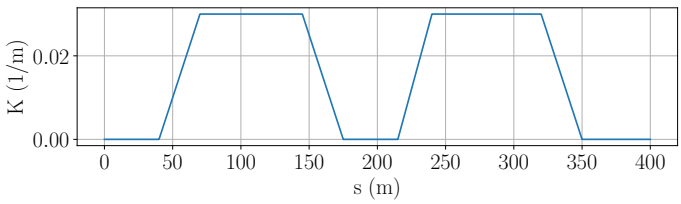


Figure 6.7: Oval trajectory test path curvature.

V_{desired}	**Method**	**Direction**	**MAE**	**Max Err.**	**Mean** V
8	HEBM	CW	**0.05**	0.26	7.97
8	KBM	CW	0.08	0.26	7.16
10	HEBM	CW	**0.072**	**0.29**	10.2
10	KBM	CW	0.13	0.59	8.33
12	HEBM	CW	**0.09**	**0.31**	12.56
12	KBM	CW	0.18	0.66	9.18
14	HEBM	CW	**0.16**	**0.48**	14.7
14	KBM	CW	0.17	0.66	9.11
16	HEBM	CW	**0.11**	**0.56**	15.86
16	KBM	CW	0.17	0.81	9.44
18	HEBM	CW	**0.2**	**0.67**	16.4
18	KBM	CW	0.33	1.77	10.85
8	HEBM	CCW	**0.05**	**0.22**	7.97
8	KBM	CCW	0.1	0.38	7.6
10	HEBM	CCW	**0.07**	**0.29**	10.2
10	KBM	CCW	0.12	0.44	7.9
12	HEBM	CCW	**0.07**	**0.4**	12.25
12	KBM	CCW	0.14	0.57	8.72
14	HEBM	CCW	**0.08**	**0.45**	14.32
14	KBM	CCW	0.2	0.69	9.91
16	HEBM	CCW	**0.15**	**0.75**	16.2
16	KBM	CCW	0.2	0.81	10.1
18	HEBM	CCW	**0.2**	**0.88**	17.8
18	KBM	CCW	0.3	1.11	11.5

Table 6.3: Oval trajectory testing results showing the ability of the proposed HEBM method to present lower lateral errors and to drive with higher velocities. (MAE and Max Error in m, V_{desired} and mean V in $\mathrm{m\,s^{-1}}$).

The presented results show that for both models, higher velocities are associated with higher errors. Though, the kinematic bicycle based planner is not comfortable with high velocities as it is not able to stick with the reference speed and to have minimal errors simultaneously. The proposed approach is able to perform maneuvers with higher velocities, sticking with the desired velocity of each case, while maintaining low lateral errors. It can be seen for example that for a $V_{\text{desired}} = 18\,\text{m}\,\text{s}^{-1}$ (CCW), the proposed approach is able to reach high velocities, performing maneuvers with lateral accelerations reaching $a_y^{\max} = 0.76g$ while keeping a maximum lateral error of 0.88 m while the kinematic bicycle approach can't keep with the desired velocity (more than 30% below) and shows higher errors. The kinematic bicycle based planner accounts for the lateral and heading errors by delivering high steering angle outputs even with increasing speeds, which is not compatible with the dynamics of the vehicle; while the hybrid extended bicycle based planner delivers more consistent outputs. The hybrid extended bicycle based planner has lower errors and higher velocities in all of the tested scenarios.

The behavior of the vehicle using each of the two planners at the highest lateral error points of the highest speed oval maneuvers can be seen in Figures 6.8 and 6.9 showing a better performance by the proposed approach that is able to stick to the reference trajectory. The showed curves are associated with a lateral acceleration of $a_y^{\max} = 0.75g$ which explains the poor performance of the kinematic bicycle model that is expected to lose validity above $a_y = 0.5g$ [Polack et al. 2017].

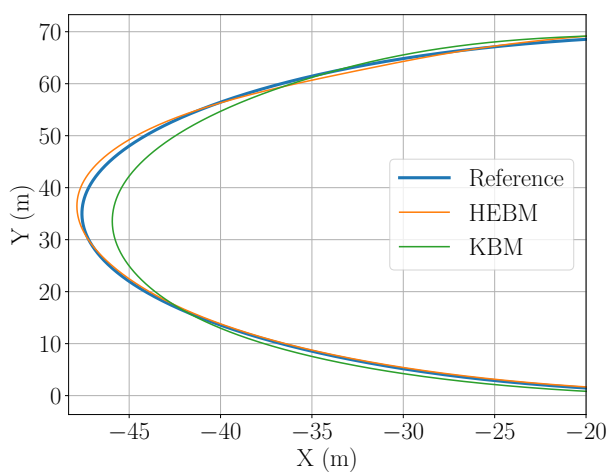


Figure 6.8: Comparison between the behavior of the vehicle subject to the KBM and HEBM planners at the highest lateral error point for the $V = 18\,\text{m}\,\text{s}^{-1}$ CW test.

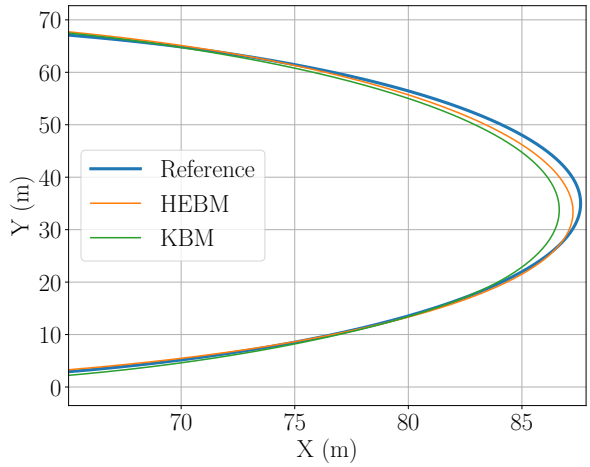


Figure 6.9: Comparison between the behavior of the vehicle subject to the KBM and HEBM planners at the highest lateral error point for the $V = 18\,\mathrm{m\,s^{-1}}$ CCW test.

The lane change trajectory is tested next.

6.5.3 Lane change trajectory

The lane change trajectory consists of running the vehicle on the reference path, presented in Chapter 3, shown in Figure 6.10 based on the ISO-3888-1 standard[3] with the curvature shown in Figure 6.11 at varying speeds starting from $X = 0$, $Y = 0$ with $V_{x,0} = V_{\mathrm{ref}}$, the other states being null except for the wheel speeds that are calculated accordingly with no initial slip.

The results of the effected test are shown in Table 6.4. The table shows that the kinematic bicycle planner is able to achieve higher velocities in comparison with the previous test, this may be due to the relaxed heading constraints of the lane change maneuver as opposed to the oval maneuver. The performance of the kinematic bicycle based planner is not accurate as opposed to the proposed model. The proposed planner is able to stick to the reference path with lower errors even at high velocities, while the kinematic bicycle based planner loses accuracy with higher velocities. The kinematic-based planner totally diverges in the last scenario.

[3] https://www.iso.org/standard/67973.html

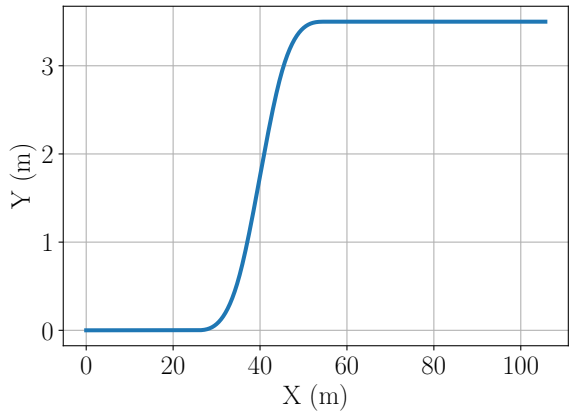


Figure 6.10: Lane change trajectory test reference path.

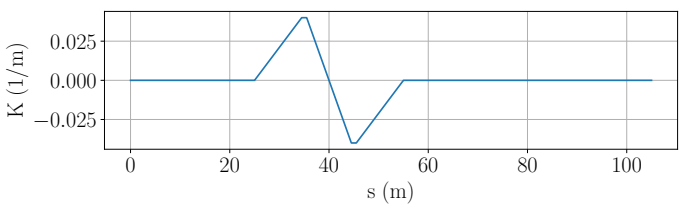


Figure 6.11: Lane change trajectory test path curvature.

V_{desired}	**Method**	**MAE**	**Max Err.**	**Mean** V
10	HEBM	**0.04**	**0.2**	10.1
10	KBM	0.09	0.4	8.47
15	HEBM	**0.05**	**0.27**	15.02
15	KBM	0.17	0.74	12.35
20	HEBM	**0.11**	**0.47**	20
20	KBM	0.38	1.12	17.93
25	HEBM	**0.2**	**0.51**	25.13
25	KBM	0.75	3.47	27.2

Table 6.4: Lane change trajectory testing results showing that the HEBM method presents lower lateral and velocity errors. (MAE and Max Error in m, V_{desired} and mean V in $\mathrm{m\,s^{-1}}$).

The highest velocity scenario is shown in Figure 6.12 showing the issues with using the kinematic bicycle based planner to plan high velocity (resulting in high dynamic) maneuvers.

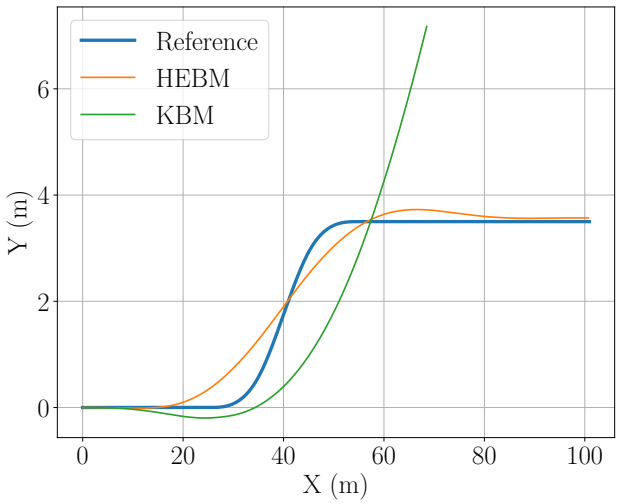


Figure 6.12: Comparison between the behavior of the vehicle subject to the KBM and HEBM planners for the highest velocity lane change maneuver. The proposed approach is able to be closer to the reference path while the KBM based approach diverges.

6.5.4 Discussion

In brief, the planner based on the hybrid extended bicycle model makes the vehicle follow the reference path in an accurate way while sticking to the reference speeds while the kinematic bicycle based planner fails to make the vehicle follow the provided reference path, especially at higher speeds, i.e. higher lateral accelerations.

6.6 Conclusion

In this chapter, the model validity issues while planning vehicle trajectories were targeted and a hybrid extended bicycle model was proposed to represent accurately the vehicle's behavior. The proposed model takes into consideration the deterministic equations of the extended bicycle model while integrating recurrent neural networks at the slip prediction level for higher accuracy. The proposed model was then integrated into an MPPI-based planning and control scheme

aiming to follow a reference path and a reference speed.

The effected tests showed a highly accurate performance by the proposed approach in comparison with the kinematic bicycle model. The inclusion of recurrent neural networks in the deterministic model allowed the resulting hybrid model to closely represent the vehicle's behavior, resulting in a more consistent planner even in higher dynamics.

Although the proposed planner was able to perform accurately in the tested scenarios at different speeds and accelerations, the validity range and the limits of the proposed solution remain an open question, especially when additional constraints, including obstacles, are added to the problem.

In a well-designed pipeline, the learned observers presented in the previous chapters would feed their estimations to the proposed planner in this chapter to result in a well-established observing-planning-controlling scheme even at the limits of handling.

Chapter 7

Conclusion and Perspectives

Contents

7.1 Conclusion **132**

7.2 Limitations and perspectives **133**

7.2.1 CNN-based observer 133

7.2.2 Learned observers applied to real vehicles 134

7.2.3 The hybrid extended bicycle model 134

7.2.4 Interchangeability of architectures 134

7.1 Conclusion

The main point that this thesis demonstrated is the ability to represent a vehicle's behavior, in normal or harsh scenarios, using learning. Whether the aim is to observe the vehicle state or to plan vehicle trajectories, the essence of the presented work lies in capturing the complex vehicle dynamics through simple neural network architectures that adapt to deal with uncertain measurements or state inputs for a wide range of scenarios. The introduced concepts enable us to surpass the limitations of deterministic vehicle models that attempt to approximate the vehicle's behavior, ensuring an accurate vehicle representation.

Accurate vehicle representation is an important factor in the functioning of autonomous vehicles. It was targeted in observing applications, where the literature tries to approximate the behavior of the vehicle through different models and observing algorithms. The presented work has set clear methods to treat noisy measurements from which the vehicle dynamics are inferred resulting in accurate state knowledge. Accurate vehicle representation plays a major role in planning applications as well, where the literature makes use of simple vehicle models with a limited validity domain, resulting in unfeasible trajectory planning. The presented work introduced a simple hybrid model with a large validity domain allowing the planning of feasible trajectories even in challenging maneuvers.

Throughout the chapters of this thesis, the above-mentioned claims were tested and proved on a specific set of tests that show the advantages of the developed methods. In summary, the main presented and developed ideas that contributed to the general purpose of this work are the following:

- **Vehicle models:** Deterministic models used to describe the behavior of the vehicle were introduced. The presentation started with complex vehicle representations to simpler ones which are used in autonomous driving applications for observing and planning purposes. The limited accuracy of deterministic models was concluded.

- **State observers:** The importance of access to key state variables of the vehicle was emphasized. To reach state estimations, model-based observers, depending on the presented vehicle models were introduced and their limits were discussed first and then demonstrated on simulated and real applications. Learned observers were proposed to deal with the limitations of model-based ones.

- **Learned observers:** The advantages of using learning techniques to gain accurate knowledge of several vehicle state variables were identified. The thesis aimed to introduce and test three observing techniques. The introduced techniques were compared to state-of-the-art model-based and

learning-based observers and showed their ability to provide accurate estimations. Two techniques were applied to a real vehicle proving the ability of learned methods to capture the parameters and dynamics of the vehicle and to deal with noisy measurements even when the vehicle is involved in high dynamic maneuvers. The use of learned observers allowed access to several state variables in a wide range of maneuvers which was not possible using state-of-the-art techniques.

- **Hybrid representation for accurate trajectory planning:** Having access to key vehicle state variables in low and high dynamic maneuvers, planning feasible vehicle trajectories was targeted. A potential solution to the issue of model validity that was interpreted throughout the different chapters was proposed and used for planning vehicle trajectories. The hybrid solution integrated recurrent neural networks into a deterministic extended bicycle model to account for slips and modeling errors in describing the behavior of a complex vehicle model. It was then implemented in an MPPI-based plan and control architecture. The approach showed its advantages over a kinematic bicycle approach in several high-speed tests.

Looking closely at the proposed solutions in this thesis, the ability to treat noisy measures using the provided techniques to deliver accurate state knowledge, and then to plan trajectories accordingly using an accurate hybrid model, allows the creation of a pipeline able to take low-quality sensor measures from which key state variables are inferred and used to accurately plan and control the vehicle for a wide range of maneuvers.

In brief, the presented techniques to either observe the state of the vehicle or plan vehicle trajectories took advantage of the integrated learning methods to result in an accurate representation of the behavior of the vehicle.

7.2 Limitations and perspectives

As mentioned in the previous section, the thesis targeted the estimation of the vehicle behavior using either measurement inputs in observing applications or state inputs in planning applications. Having presented the advantages of the proposed techniques, the limitations and perspectives are presented in this section.

7.2.1 CNN-based observer

The CNN observer presented in Chapter 4 was able to outperform the EKF-based observer in high noise domains. The mentioned work was effected on a simulator based on the kinematic bicycle model with Gaussian noise simulating the behavior of the sensor. The work was not transferred to real applications due to the limitations of the dataset collected in TU Braunschweig.

Investigating the performance of the solution in real applications would give a real insight into its effectiveness when dealing with actual sensors representing the position of a real vehicle.

Also, the proposed method takes as inputs the absolute coordinates of the vehicle which would create problems in large distances. Using relative distance changes would be more suitable for real applications.

7.2.2 Learned observers applied to real vehicles

The learned and hybrid observers applied to the Stadtpilot vehicle in Chapter 5 were able to provide accurate estimations in low and high dynamic maneuvers.

The robustness of these observers to significant changes in the mass of the vehicle, the friction coefficient of the road, and other varying parameters of the vehicle should be investigated. Notably, the work in [Zhang et al. 2023] targeted mass estimation while the work in [Spielberg et al. 2022] assessed the behavior of a learned model to varying friction on the road. The robustness of the developed approaches to changes in internal or external conditions is a question to be answered.

Also, with the rise of attention-based techniques and the ongoing direction of learning-based research towards transformers, the role that transformers could play in state observation and vehicle behavior representation in general, is an interesting question that should be explored. Is "attention all you need" for vehicle state estimation?

7.2.3 The hybrid extended bicycle model

The hybrid extended bicycle model was proposed in Chapter 6 aiming to plan feasible trajectories for the vehicle to follow despite the harshness of the maneuver. The proposed architecture showed accurate results and outperformed the kinematic bicycle model which is used in several literature applications.

The ability of the developed model to integrate into a real vehicle should be explored. Although a complex four-wheel vehicle model with a Pacejka tire model was used at the base of the simulator, behavioral differences would still exist in comparison with real vehicles. Also, using a simulator-based approach eliminates the delays involved in dealing with real systems. These delays could impose safety threats if the developed solution is not prepared to deal with them.

7.2.4 Interchangeability of architectures

This thesis proposed multiple strategies to estimate the dynamics of the vehicle. A question that arises is the ability to interchange architectures to reach similar goals. In other words, as the vehicle velocity observer estimates the longitudinal

and lateral velocities, is it able to compete with the side-slip angle observer? Also, can the hybrid extended bicycle model be implemented to achieve wheel slip angle observations, or vehicle velocity and yaw rate observations?

These questions would require more tests and comparisons but may conclude the limitations of fully learned and hybrid techniques.

Finally, a question that arises for all of the proposed solutions is their transferability to other vehicles and the difficulty of employing any of them to another vehicle with different properties.

Bibliography

Abbas, M. A., Milman, R., and Eklund, J. M. (2017). Obstacle avoidance in real time with Nonlinear Model Predictive Control of autonomous vehicles. *2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pages 1–6.

Altché, F., Polack, P., and de La Fortelle, A. (2017). A Simple Dynamic Model for Aggressive, Near-Limits Trajectory Planning.

Bacha, A., Bauman, C., Faruque, R., Fleming, M., Terwelp, C., Reinholtz, C., Hong, D., Wicks, A., Alberi, T., Anderson, D., Cacciola, S., Currier, P., Dalton, A., Farmer, J., Hurdus, J., Kimmel, S., King, P., Taylor, A., Covern, D. V., and Webster, M. (2008). Odin: Team VictorTango's entry in the DARPA Urban Challenge. *Journal of Field Robotics*, 25(8):467–492.

Baffet, G., Charara, A., and Lechner, D. (2007). Experimental evaluation of a sliding mode observer for tire-road forces and an extended Kalman filter for vehicle sideslip angle. In *2007 46th IEEE Conference on Decision and Control*, pages 3877–3882, New Orleans, LA, USA. IEEE.

Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. (2011). Algorithms for Hyper-Parameter Optimization. In Shawe-Taylor, J., Zemel, R., Bartlett, P., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Bergstra, J. and Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. *J. Mach. Learn. Res.*, 13(null):281–305. Publisher: JMLR.org.

Bohren, J., Foote, T., Keller, J., Kushleyev, A., Lee, D., Stewart, A., Vernaza, P., Derenick, J., Spletzer, J., and Satterfield, B. (2008). Little Ben: The Ben Franklin Racing Team's entry in the 2007 DARPA Urban Challenge. *Journal of Field Robotics*, 25(9):598–614.

Bonnabel, S. (2007). Left-invariant extended Kalman filter and attitude estimation. In *2007 46th IEEE Conference on Decision and Control*, pages 1027–1032, New Orleans, LA, USA. IEEE.

Broggi, A., Medici, P., Zani, P., Coati, A., and Panciroli, M. (2012). Autonomous vehicles control in the VisLab Intercontinental Autonomous Challenge. *Annual Reviews in Control*, 36(1):161–171.

Burger, C., Fischer, J., Bieder, F., Tas, O. S., and Stiller, C. (2022). Interaction-Aware Game-Theoretic Motion Planning for Automated Vehicles using Bi-level Optimization. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3978–3985, Macau, China. IEEE.

Camacho, E. F., Bordons, C., and Normey-Rico, J. E. (2003). Model predictive control springer, Berlin, 1999, ISBN 3540762418, 280 pages. *International Journal of Robust and Nonlinear Control*, 13(11):1091–1093.

Canudas De Wit, C. and Slotine, J.-J. (1989). Sliding Observers for Robot Manipulators. *IFAC Proceedings Volumes*, 22(3):379–384.

Cardoso, V., Oliveira, J., Teixeira, T., Badue, C., Mutz, F., Oliveira-Santos, T., Veronese, L., and De Souza, A. (2016). A Model-Predictive Motion Planner for the IARA Autonomous Car.

Chen, B.-C. and Hsieh, F.-C. (2008). Sideslip angle estimation using extended Kalman filter. *Vehicle System Dynamics*, 46(sup1):353–364.

Cherouat, H., Braci, M., and Diop, S. (2005). Vehicle velocity, side slip angles and yaw rate estimation. In *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005. ISIE 2005.*, pages 349–354, Dubrovnik, Croatia. IEEE.

Choi, G., Park, J., Shlezinger, N., Eldar, Y. C., and Lee, N. (2023). Split-KalmanNet: A Robust Model-Based Deep Learning Approach for State Estimation. *IEEE Transactions on Vehicular Technology*, pages 1–6.

Choi, J. Y., Hong, S. J., Park, K. T., Yoo, W. S., and Lee, M. H. (2002). Lateral control of autonomous vehicle by yaw rate feedback. *KSME International Journal*, 16(3):338–343.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. Publisher: arXiv Version Number: 1.

Coulter, R. C. (1992). Implementation of the Pure Pursuit Path Tracking Algorithm. *undefined*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

Ding, N. and Taheri, S. (2010). A Modified Dugoff Tire Model for Combined-slip Forces. *Tire Science and Technology*, 38(3):228–244.

Drakunov, S. and Utkin, V. (1995). Sliding mode observers. Tutorial. In *Proceedings of 1995 34th IEEE Conference on Decision and Control*, volume 4, pages 3376–3378, New Orleans, LA, USA. IEEE.

Dugoff, H. (1969). *Tire performance characteristics affecting vehicle response to steering and braking control inputs. Final report.*

Elkaim, G. H., Lizarraga, M., and Pederseny, L. (2008). Comparison of low-cost GPS/INS sensors for Autonomous Vehicle applications. In *2008 IEEE/ION Position, Location and Navigation Symposium*, pages 1133–1144.

Escoriza, A. L., Revach, G., Shlezinger, N., and van Sloun, R. J. G. (2021). Data-Driven Kalman-Based Velocity Estimation for Autonomous Racing. In *2021 IEEE International Conference on Autonomous Systems (ICAS)*, pages 1–5.

Gao, X. and Yu, Z. (2010). Nonlinear Estimation of Vehicle Sideslip Angle Based on Adaptive Extended Kalman Filter. SAE Technical Paper 2010-01-0117, SAE Technical Paper, Warrendale, PA.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In Teh, Y. W. and Titterington, M., editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy. PMLR.

Godbole, D., Hagenmeyer, V., Sengupta, R., and Swaroop, D. (1997). Design of emergency manoeuvres for automated highway system: obstacle avoidance problem. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 5, pages 4774–4779, San Diego, CA, USA. IEEE.

Gonzalez, D., Perez, J., Milanes, V., and Nashashibi, F. (2016). A Review of Motion Planning Techniques for Automated Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 17(4):1135–1145.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. Adaptive Computation and Machine Learning series. MIT Press.

Graber, T., Lupberger, S., Unterreiner, M., and Schramm, D. (2019). A Hybrid Approach to Side-Slip Angle Estimation With Recurrent Neural Networks and Kinematic Vehicle Models. *IEEE Transactions on Intelligent Vehicles*, 4(1):39–47.

Guo, H., Ma, B., Ma, Y., and Chen, H. (2016). Modular scheme for vehicle tire forces and velocities estimation based on sliding mode observer. In *2016 Chinese Control and Decision Conference (CCDC)*, pages 5661–5666, Yinchuan, China. IEEE.

Guo, K. and Ren, L. (1999). A Unified Semi-Empirical Tire Model with Higher-Accuracy and Less Parameters. pages 1999–01–0785.

Hac, A. and Simpson, M. D. (2000). Estimation of Vehicle Side Slip Angle and Yaw Rate. pages 2000–01–0696.

Hart, P., Nilsson, N., and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hrgetic, M., Deur, J., Ivanovic, V., and Tseng, E. (2014). Vehicle Sideslip Angle EKF Estimator based on Nonlinear Vehicle Dynamics Model and Stochastic Tire Forces Modeling. *SAE International Journal of Passenger Cars - Mechanical Systems*, 7(1):86–95.

Kalman, R. (1960a). On the general theory of control systems. *IFAC Proceedings Volumes*, 1(1):491–502.

Kalman, R. E. (1960b). A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45.

Katriniok, A. and Abel, D. (2016). Adaptive EKF-Based Vehicle State Estimation With Online Assessment of Local Observability. *IEEE Transactions on Control Systems Technology*, 24(4):1368–1381.

Kiencke, U. and Daiß, A. (1997). Observation of lateral vehicle dynamics. *Control Engineering Practice*, 5(8):1145–1150.

Kim, M.-s., Kim, B.-j., Kim, C.-i., So, M.-h., Lee, G.-s., and Lim, J.-h. (2018). Vehicle Dynamics and Road Slope Estimation based on Cascade Extended Kalman

Filter. In *2018 International Conference on Information and Communication Technology Robotics (ICT-ROBOT)*, pages 1–4.

Kuwata, Y., Teo, J., Fiore, G., Karaman, S., Frazzoli, E., and How, J. (2011). Real-Time Motion Planning With Applications to Autonomous Urban Driving. *IEEE*.

LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press, Cambridge, U.K.

LaValle, S. M. and Kuffner, J. J. (2001). Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, 20(5):378–400.

Le Cun, Y. and Fogelman-Soulié, F. (1987). Modèles connexionnistes de l'apprentissage. *Intellectica. Revue de l'Association pour la Recherche Cognitive*, 2(1):114–143.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551.

Lee, S.-H., Son, Y., Kang, C. M., and Chung, C. C. (2013). Slip Angle Estimation: Development and Experimental Evaluation. *IFAC Proceedings Volumes*, 46(10):286–291.

Lenain, R., Thuilot, B., Cariou, C., and Martinet, P. (2003). Adaptive control for car like vehicles guidance relying on RTK GPS: rejection of sliding effects in agricultural applications. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 1, pages 115–120, Taipei, Taiwan. IEEE.

Liu, J. and Guo, G. (2021). Vehicle Localization During GPS Outages With Extended Kalman Filter and Deep Learning. *IEEE Transactions on Instrumentation and Measurement*, 70:1–10.

Liu, W., Liu, W., Ding, H., and Guo, K. (2012). Side-slip angle estimation for vehicle Electronic Stability Control based on sliding mode observer. In *Proceedings of 2012 International Conference on Measurement, Information and Control*, pages 992–995, Harbin, China. IEEE.

Ma, B., Guo, H., Zhang, H., Chen, H., and Sun, Z. (2017). Estimation of road grade and vehicle velocity for autonomous driving vehicle. In *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, pages 4621–4626, Beijing. IEEE.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133.

Ming Feng Hsieh and Ozguner, U. (2008). A parking algorithm for an autonomous vehicle. In *2008 IEEE Intelligent Vehicles Symposium*, pages 1155–1160, Eindhoven, Netherlands. IEEE.

Mitschke, M. (1990). *Dynamik der Kraftfahrzeuge*. Springer Berlin Heidelberg, Berlin, Heidelberg.

Mohamed, I. S., Allibert, G., and Martinet, P. (2021). Sampling-Based MPC for Constrained Vision Based Control. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3753–3758, Prague, Czech Republic. IEEE.

Murphy, R. (2000). *Introduction to AI robotics*. Intelligent robotics and autonomous agents. MIT Press, Cambridge, Mass.

Pacejka, H. B. (2006). *Tyre and Vehicle Dynamics*. Butterworth-Heinemann.

Pacejka, H. B. and Besselink, I. J. M. (1997). Magic Formula Tyre Model with Transient Properties. *Vehicle System Dynamics*, 27(sup001):234–249.

Park, J.-M., Kim, D.-W., Yoon, Y.-S., Kim, H. J., and Yi, K.-S. (2009). Obstacle avoidance of autonomous vehicles based on model predictive control. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 223(12):1499–1516.

Piyabongkarn, D., Rajamani, R., Grogg, J., and Lew, J. (2009). Development and Experimental Evaluation of a Slip Angle Estimator for Vehicle Stability Control. *IEEE Transactions on Control Systems Technology*, 17(1):78–88.

Polack, P. (2018). *Consistency and stability of hierarchical planning and control systems for autonomous driving*. PhD thesis.

Polack, P., Altché, F., D'Andrea-Novel, B., and de La Fortelle, A. (2018). Guaranteeing Consistency in a Motion Planning and Control Architecture Using a Kinematic Bicycle Model. In *2018 Annual American Control Conference (ACC)*, pages 3981–3987.

Polack, P., Altché, F., d'Andréa Novel, B., and de La Fortelle, A. (2017). The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 812–818.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., and others (2018). Improving language understanding by generative pre-training. Publisher: OpenAI.

Rajamani, R. (2012). Lateral Vehicle Dynamics. In Rajamani, R., editor, *Vehicle Dynamics and Control*, Mechanical Engineering Series, pages 15–46. Springer US, Boston, MA.

Reina, G. and Messina, A. (2019). Vehicle dynamics estimation via augmented Extended Kalman Filtering. *Measurement*, 133:383–395.

Revach, G., Shlezinger, N., van Sloun, R. J. G., and Eldar, Y. C. (2021). Kalmannet: Data-Driven Kalman Filtering. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3905–3909, Toronto, ON, Canada. IEEE.

Roberts, A., Raffel, C., Lee, K., Matena, M., Shazeer, N., Liu, P. J., Narang, S., Li, W., and Zhou, Y. (2019). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Technical report, Google.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Savitzky, A. and Golay, M. J. E. (1964). Smoothing and Differentiation of Data by Simplified Least Squares Procedures. *Analytical Chemistry*, 36(8):1627–1639.

Sebsadji, Y., Glaser, S., Mammar, S., and Dakhlallah, J. (2008). Road slope and vehicle dynamics estimation. In *2008 American Control Conference*, pages 4603–4608, Seattle, WA. IEEE.

Snoek, J., Larochelle, H., and Adams, R. P. (2012). Practical bayesian optimization of machine learning algorithms. *Advances in neural information processing systems*, 25.

Song, R. and Fang, Y. (2021). Vehicle state estimation for INS/GPS aided by sensors fusion and SCKF-based algorithm. *Mechanical Systems and Signal Processing*, 150:107315.

Song, S., Chun, M. C. K., Huissoon, J., and Waslander, S. L. (2014). Pneumatic trail based slip angle observer with Dugoff tire model. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 1127–1132.

Spentzas, K. N., Alkhazali, I., and Demic, M. (2001). Kinematics of four-wheel-steering vehicles. *Forschung im Ingenieurwesen*, 66(5):0211–0216.

Spielberg, N. A., Brown, M., and Gerdes, J. C. (2022). Neural Network Model Predictive Motion Control Applied to Automated Driving With Unknown Friction. *IEEE Transactions on Control Systems Technology*, 30(5):1934–1945.

Srinivasan, S., Sa, I., Zyner, A., Reijgwart, V., Valls, M. I., and Siegwart, R. (2020). End-to-End Velocity Estimation For Autonomous Racing.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., Diebel, J., Fong, P., Gale, J., Halpenny, M., Hoffmann, G., Lau, K., Oakley, C., Palatucci, M., Pratt, V., Stang, P., Strohband, S., Dupont, C., Jendrossek, L.-E., Koelen, C., Markey, C., Rummel, C., van Niekerk, J., Jensen, E., Alessandrini, P., Bradski, G., Davies, B., Ettinger, S., Kaehler, A., Nefian, A., and Mahoney, P. (2006). Stanley: The robot that won the DARPA Grand Challenge. *Journal of Field Robotics*, 23(9):661–692.

Tielking, J. and Mital, N. K. (1974). A comparative evaluation of five traction tire models [R]. *Highway Safety Research Institute Report*, pages 1–27.

van Aalst, S., Naets., F., Boulkroune, B., Nijs, W. D., and Desmet, W. (2018). An Adaptive Vehicle Sideslip Estimator for Reliable Estimation in Low and High Excitation Driving. *IFAC-PapersOnLine*, 51(9):243–248.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is All you Need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Wan, E. and Van Der Merwe, R. (2000). The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No.00EX373)*, pages 153–158, Lake Louise, Alta., Canada. IEEE.

Wang, W., Yuan, L., Tao, S.-J., Zhang, W., and Su, T. (2010). Estimation of vehicle side slip angle in nonlinear condition based on the state feedback observer. *2010 IEEE International Conference on Automation and Logistics*.

Wenzel, T. A., Burnham, K. J., Blundell, M. V., and Williams, R. A. (2006). Dual extended Kalman filter for vehicle state and parameter estimation. *Vehicle System Dynamics*, 44(2):153–171.

Williams, G., Drews, P., Goldfain, B., Rehg, J. M., and Theodorou, E. A. (2016). Aggressive driving with model predictive path integral control. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1433–1440, Stockholm. IEEE.

Williams, G., Drews, P., Goldfain, B., Rehg, J. M., and Theodorou, E. A. (2018). Information-Theoretic Model Predictive Control: Theory and Applications to Autonomous Driving. *IEEE Transactions on Robotics*, 34(6):1603–1622.

Yin, Z., Guo, H., Wang, F., Chen, H., and Lv, K. (2017). Design for vehicle velocity estimation based on reduced-order observer. In *2017 Chinese Automation Congress (CAC)*, pages 6976–6981, Jinan. IEEE.

Zhang, B., Zhao, W., Zou, S., Zhang, H., and Luan, Z. (2021). A Reliable Vehicle Lateral Velocity Estimation Methodology Based on SBI-LSTM During GPS-Outage. *IEEE Sensors Journal*, 21(14):15485–15495.

Zhang, H., Yang, Z., Xiong, H., Zhu, T., Long, Z., and Wu, W. (2023). Transformer Aided Adaptive Extended Kalman Filter for Autonomous Vehicle Mass Estimation. *Processes*, 11(3):887.

Zhao, L.-H., Liu, Z.-Y., and Chen, H. (2009). Sliding mode observer for vehicle velocity estimation with road grade and bank angles adaptation. In *2009 IEEE Intelligent Vehicles Symposium*, pages 701–706, Xi'an, China. IEEE.

Zhao, L.-H., Liu, Z.-Y., and Chen, H. (2011). Design of a Nonlinear Observer for Vehicle Velocity Estimation and Experiments. *IEEE Transactions on Control Systems Technology*, 19(3):664–672.

Zhao Linhui, Liu Zhiyuan, and Chen Hong (2008). Vehicle state and friction force estimation using nonlinear observer strategies. In *2008 27th Chinese Control Conference*, pages 667–671, Kunming, China. IEEE.

Zhou and Chellappa (1988). Computation of optical flow using a neural network. In *IEEE International Conference on Neural Networks*, pages 71–78 vol.2, San Diego, CA, USA. IEEE.

Ziegler, J. and Werling, M. (2008). Navigating car-like robots in unstructured environments using an obstacle sensitive cost function. In *2008 IEEE Intelligent Vehicles Symposium*, pages 787–791, Eindhoven, Netherlands. IEEE.

Zong, C., Hu, D., and Zheng, H. (2013). Dual extended Kalman filter for combined estimation of vehicle state and road friction. *Chinese Journal of Mechanical Engineering*, 26(2):313–324.

Åström, K. J. and Hägglund, T. (1995). *PID Controllers: Theory, Design, and Tuning*. ISA - The Instrumentation, Systems and Automation Society.

RÉSUMÉ

Les progrès des véhicules autonomes représentent une avancée significative dans la recherche de modes de transport plus sûrs et plus fiables. Atteindre l'autonomie complète des véhicules est l'objectif principal des chercheurs dans ce domaine au cours des dernières années. L'autonomie complète exige une représentation précise de la dynamique du véhicule à travers les différents composants de son architecture, afin d'assurer le fonctionnement dans une large gamme de scénarios. Pour atteindre cet objectif, cette thèse introduit la notion d'apprentissage pour les observateurs et les planificateurs de véhicules.
Grâce à l'intégration de techniques d'hybridation entre modélisation et apprentissage, notre objectif est d'améliorer la capacité du véhicule à observer son état. L'obtention d'une connaissance précise de l'état est essentielle pour les couches de planification et de contrôle qui dépendent de cette information. Les techniques d'observation proposées sont testées sur des applications de véhicules réels, ce qui prouve la capacité des méthodes proposées à réaliser des observations précises dans des scénarios réels, même aux limites de la manipulation du véhicule. Les méthodes proposées présentent des avantages significatifs par rapport aux méthodes de l'état de l'art.
Après avoir obtenu des observations précises de l'état du véhicule, nous proposons, dans un deuxième temps, un modèle hybride simple et précis. Ce modèle décrit précisément le comportement du véhicule, ce qui permet de développer un planificateur capable de générer des trajectoires réalisables, même dans des scénarios très dynamiques. Un schéma de planification et de contrôle basés sur la technique MPPI sont proposés et testés pour diverses manœuvres. En comparant notre approche au modèle cinématique de bicyclette couramment utilisé dans les applications de planification, nos résultats démontrent la supériorité de la méthode proposée. Notamment, le planificateur utilisant notre modèle hybride garantit un comportement plus sûr et plus précis du véhicule.
Cette thèse démontre les capacités des architectures de réseaux neuronaux appris et hybrides proposées à représenter avec précision la dynamique complexe du véhicule. Grâce à des expériences sur des véhicules simulés et réels, les méthodes proposées prouvent leur capacité à surpasser les méthodes de pointe dans les applications d'observation et de planification.

MOTS CLÉS

Contrôle, Observateur, Planificateur, Apprentissage Automatique, Véhicule

ABSTRACT

The advancement of autonomous vehicles represents a significant leap forward in the pursuit of safer and more reliable modes of transportation. Reaching full autonomy in vehicles has become the central focus of researchers and experts in the field in the last decade. Full autonomy demands precise representation of the vehicle's dynamics across various components within its architecture, to ensure the operation in a wide range of scenarios. To achieve this purpose, this thesis integrates the notion of learning to vehicle observers and planners.
Through the integration of hybrid and learned techniques, our objective is to greatly enhance the vehicle's capacity to accurately observe its state. Achieving precise state knowledge is critical as both the planning and control layers rely on this information. We test our observing techniques on real vehicle applications proving the ability of the proposed methods to achieve accurate observations in real-life scenarios, even at the limits of handling of the vehicle. The proposed methods present significant advantages over state-of-the-art methods.
After achieving accurate state observations, we propose a simple yet accurate hybrid model in the second stage. This model precisely describes the vehicle's behavior, allowing the development of a planner that can generate feasible trajectories, even in high-dynamic scenarios. An MPPI-based plan and control scheme is proposed and thoroughly tested across various maneuvers. Comparing our approach to the commonly used kinematic bicycle model in planning applications, our results clearly demonstrate the superiority of the proposed method. Notably, the planner utilizing our hybrid model ensures safer and more precise vehicle behavior.
This thesis demonstrates the capabilities of the proposed learned and hybrid neural network architectures in accurately representing the complex dynamics of the vehicle. Through simulated and real vehicle experiments, the proposed methods prove their ability to outperform state-of-the-art methods in observing and planning applications.

KEYWORDS

Control, Observer, Learning-based control, Planner, Machine Learning, Vehicle